algorithms

MDPI

Article

On the Descriptive Complexity of Color Coding †

Max Bannach * and Till Tantau *

Institute for Theoretical Computer Science, Universität zu Lübeck, 23562 Lübeck, Germany
* Correspondence: bannach@tcs.uni-luebeck.de (M.B.); tantau@tcs.uni-luebeck.de (T.T.)
† This paper is an extended version of our paper published in STACS 2019.

Abstract: Color coding is an algorithmic technique used in parameterized complexity theory to detect "small" structures inside graphs. The idea is to derandomize algorithms that first randomly color a graph and then search for an easily-detectable, small color pattern. We transfer color coding to the world of descriptive complexity theory by characterizing—purely in terms of the syntactic structure of describing formulas—when the powerful second-order quantifiers representing a random coloring can be replaced by equivalent, simple first-order formulas. Building on this result, we identify syntactic properties of first-order quantifiers that can be eliminated from formulas describing parameterized problems. The result applies to many packing and embedding problems, but also to the long path problem. Together with a new result on the parameterized complexity of formula families involving only a fixed number of variables, we get that many problems lie in FPT just because of the way they are commonly described using logical formulas.

Keywords: color coding; descriptive complexity; fixed-parameter tractability; quantifier elimination; para-AC^0

Citation: Bannach, M.; Tantau, T. On the Descriptive Complexity of Color Coding. *Algorithms* **2021**, *14*, 96. https://doi.org/10.3390/a14030096

Academic Editor: Arne Meier

Received: 29 January 2021
Accepted: 16 March 2021
Published: 19 March 2021

1. Introduction

Descriptive complexity provides a powerful link between logic and complexity theory: We use a logical formula to describe a problem and can then infer the computational complexity of the problem just from the syntactic structure of the formula. As a striking example, Fagin's Theorem [1] tells us that 3-colorability lies in NP just because its describing formula ("there exist three colors such that all adjacent vertex pairs have different colors") is an existential second-order formula. In the context of fixed-parameter tractability theory, methods from descriptive complexity are also used a lot—but commonly to show that problems are difficult. For instance, the A- and W-hierarchies are defined in logical terms [2], but their hard problems are presumably "beyond" the class FPT of fixed-parameter tractable problems.

The methods of descriptive complexity are only rarely used to show that problems are in FPT. More precisely, the syntactic structure of the natural logical descriptions of standard parameterized problems found in textbooks are not known to imply that the problems lie in FPT—even though this is known to be the case for many of them. To appreciate the underlying difficulties, consider the following three parameterized problems (the prefix "p" stand for "parameterized" and its index names the parameter): p_k-MATCHING, p_k-TRIANGLE-PACKING, and p_k-CLIQUE. In each case, we are given an undirected graph as input and a number k and we are then asked whether the graph contains k vertex-disjoint edges (a size-k matching), k vertex-disjoint triangles, or a clique of size k, respectively. The problems are known to have widely different complexities (maximal matchings can actually be found in polynomial time, triangle packing lies at least in FPT, while finding cliques is W[1]-complete) but very similar logical descriptions:

$$\alpha_k = \exists x_1 \cdots \exists x_{2k} \big(\bigwedge_{i \neq j} x_i \neq x_j \wedge \bigwedge_{i=1}^{k} Ex_{2i-1}x_{2i} \big), \tag{1}$$

$$\beta_k = \exists x_1 \cdots \exists x_{3k} \big(\bigwedge_{i \neq j} x_i \neq x_j \wedge \bigwedge_{i=1}^{k} (Ex_{3i-2}x_{3i-1} \wedge Ex_{3i-2}x_{3i} \wedge Ex_{3i-1}x_{3i}) \big), \tag{2}$$

$$\gamma_k = \exists x_1 \cdots \exists x_k \big(\bigwedge_{i \neq j} x_i \neq x_j \wedge \bigwedge_{i \neq j} Ex_i x_j \big). \tag{3}$$

The family $(\alpha_k)_{k \in \mathbb{N}}$ of formulas is clearly a natural "slicewise" description of the matching problem: A graph $\mathcal{G}$ has a size-k matching if and only if $\mathcal{G} \models \alpha_k$. The families $(\beta_k)_{k \in \mathbb{N}}$ and $(\gamma_k)_{k \in \mathbb{N}}$ are natural parameterized descriptions of the triangle packing and the clique problems, respectively. Well-known results on the descriptive complexity of parameterized problems allow us to infer [2] from the above descriptions that all three problems lie in W[1], but offer no hint why the first two problems actually lie in the class FPT—syntactically the clique problem arguably "looks like the easiest one" when in fact it is semantically the most difficult one. The results of this paper will remedy this mismatch between syntactic and semantic complexity: We will show that the syntactic structures of the formulas α_k and β_k imply membership of p$_k$-MATCHING and p$_k$-TRIANGLE-PACKING in FPT. In particular, we will identify the non-obvious syntactic properties that make α_k and β_k "easier" than γ_k.

The road to deriving the computational complexity of parameterized problems just from the syntactic properties of slicewise first-order descriptions involves three major steps: First, a characterization of when the color coding technique is applicable in terms of syntactic properties of second-order quantifiers. Second, an exploration of how these results on second-order formulas apply to first-order formulas, leading to the notion of strong and weak quantifiers and to an elimination theorem for weak quantifiers. Third, we add a new characterization to the body of known characterizations of how classes like FPT can be characterized in a slicewise fashion by logical formulas.

Our Contributions I: A Syntactic Characterization of Color Coding.

Consider once more the triangle packing problem, where we are asked whether an undirected graph G contains k vertex-disjoint triangles. While this problem is known to be hard, it becomes almost trivial if we change it slightly: Suppose someone colored the vertices of the graph and our job is just to determine whether there are a red triangle, a green triangle, a blue triangle, a yellow triangle, and so on for k different colors. Clearly, this is a now a very easy problem and if we do, indeed, find k triangles having k different colors, we have found k vertex-disjoint triangles.

The ingenious idea behind the color coding technique of Alon, Yuster, and Zwick [3] is to reduce the hard triangle packing problem to the much simpler colored version by simply randomly coloring the graph. Of course, even if there are k disjoint triangles, we will most likely not color them monochromatically and differently, but the probability of "getting lucky" is nonzero (at least k^{-3k}) and depends only on the parameter k. Even better, Alon et al. point out that one can derandomize the coloring easily by using universal hash functions to color each vertex with its hash value.

Applying this idea in the setting of descriptive complexity was recently pioneered by Chen et al. [4]. Transferred to the triangle packing problem, their argument would roughly be: "Testing for each color i, whether there is a monochromatic triangle of color i, can be done in first-order logic using something like $\bigwedge_{i=1}^{k} \exists x \exists y \exists z (Exy \wedge Eyz \wedge Exz \wedge C_i x \wedge C_i y \wedge C_i z)$. Next, instead of testing whether x has color i using the formula $C_i x$, we can test whether x gets hashed to i by a hash function. Finally, since computing appropriate universal hash functions only involves addition and multiplication, we can express the derandomized algorithm using an arithmetic first-order formula of low quantifier rank." Phrased differently, Chen et al. would argue that $\bigwedge_{i=1}^{k} \exists x \exists y \exists z (Exy \wedge Eyz \wedge Exz \wedge C_i x \wedge C_i y \wedge C_i z)$ together with the requirement that the C_i are pairwise disjoint is (ignoring some details) equivalent to $\delta_k = \exists p \exists q \bigwedge_{i=1}^{k} \exists x \exists y \exists z (Exy \wedge Eyz \wedge Exz \wedge \text{HASH}_k(x,p,q) = i \wedge \text{HASH}_k(y,p,q) = i \wedge \text{HASH}_k(z,p,q) = i)$, where $\text{HASH}_k(x,p,q) = i$ is a formula that is

true when "x is hashed to i by a member of a universal family of hash functions indexed by q and p."

The family $(\delta_k)_{k\in\mathbb{N}}$ may seem rather technical and, indeed, its importance becomes visible only in conjunction with another result by Chen et al. [4]: They show that a parameterized problem lies in para-AC^0, one of the smallest "sensible" subclasses of FPT, if it can be described by a family $(\phi_k)_{k\in\mathbb{N}}$ of FO$[+,\times]$ formulas of bounded quantifier rank such that the finite models of ϕ_k are exactly the elements of the kth slice of the problem. Since the triangle packing problem can be described in this way via the family $(\delta_k)_{k\in\mathbb{N}}$ of formulas, all of which have a quantifier rank 5 plus the constant number of quantifiers used to express the arithmetics in the formulas $\text{HASH}_k(x,p,q) = i$, we get p_k-TRIANGLE-PACKING $\in$ FPT.

Clearly, this beautiful idea cannot work in all situations: If it also worked for the formula mentioned earlier expressing 3-colorability, 3-colorability would be first-order expressible, which is known to be impossible. Our first main contribution is a syntactic characterization of when the color coding technique is applicable, that is, of why color coding works for triangle packing but not for 3-colorability: For triangle packing, the colors C_i are applied to variables only inside existential scopes ("$\exists x\exists y\exists z$") while for 3-colorability the colors R, G, and B are also applied to variables inside universal scopes ("for all adjacent vertices"). In general, see Theorem 2 for the details, we show that a second-order quantification over an arbitrary number of disjoint colors C_i can be replaced by a fixed number of first-order quantifiers whenever none of the C_i is used in a universal scope.

Our Contributions II: New First-Order Quantifier Elimination Rules.

The "purpose" of the colors C_i in the formulas $\bigwedge_{i=1}^{k}\exists x\exists y\exists z(Exy \wedge Eyz \wedge Exz \wedge C_i x \wedge C_i y \wedge C_i z)$ is not that the three vertices of a triangle get a particular color, but just that they get a color different from the color of all other triangles. Indeed, our "real" objective in these formulas is to ensure that the vertices of a triangle are distinct from the vertices in the other triangles—and giving vertices different colors is "just a means" of ensuring this.

In our second main contribution we explore this idea further: If the main (indeed, the only) use of colors in the context of color coding is to ensure that certain vertices are different, let us do away with colors and instead focus on the notion of distinctness. To better explain this idea, consider the following family, also describing triangle packing, where the only change is that we now require (a bit superfluously) that even the vertices inside a triangle get different colors: $\bigwedge_{j=1}^{k}\exists x\exists y\exists z(Exy \wedge Eyz \wedge Exz \wedge C_{3j-2}x \wedge C_{3j-1}y \wedge C_{3j}z)$. Observe that each C_i is now applied to exactly one variable (x, y, or z in one of the many literals) and the only "effect" that all these applications have is to ensure that the vertices to which these variables get bound are different. In particular, the formula is equivalent to

$$\exists x_1 \cdots \exists x_{3k} \bigwedge_{i\neq j} x_i \neq x_j \wedge \bigwedge_{j=1}^{k}\exists x\exists y\exists z(Exy \wedge Eyz \wedge Exz \wedge x_{3j-2} = x \wedge x_{3j-1} = y \wedge x_{3j} = z) \tag{4}$$

and these formulas are clearly equivalent to the almost identical formulas from (2).

In a sense, in (4) the many existential quantifiers $\exists x_i$ and the many $x_i \neq x_j$ literals come "for free" from the color coding technique, while $\exists x$, $\exists y$, and $\exists z$ have nothing to do with color coding. Our key observation is a syntactic property that tells us whether a quantifier comes "for free" in this way (we will call it weak) or not (we will call it strong): Definition 3 states (essentially) that weak quantifiers have the form $\exists x(\phi)$ such that x is not free in any universal scope of ϕ and x is used in at most one literal that is not of the form $x \neq y$. To make weak quantifiers easier to spot, we mark the variables they bind with a dot (note that this is just a "syntactic hint to the reader" without any semantic meaning). Formulas (4) now read $\exists \dot{x}_1 \cdots \exists \dot{x}_{3k} \bigwedge_{i\neq j} \dot{x}_i \neq \dot{x}_j \wedge \bigwedge_{j=1}^{k}\exists x\exists y\exists z(Exy \wedge Exz \wedge Eyz \wedge \dot{x}_{3j-2} = x \wedge \dot{x}_{3j-1} = y \wedge \dot{x}_{3j} = z)$. Observe that x, y, and z are not weak since each is used in three literals that are not inequalities.

We show in Theorem 4 that each ϕ is equivalent to a ϕ' whose quantifier rank depends only on the strong quantifier rank of ϕ (meaning that we ignore the weak quantifiers) and whose number of variables depends only on the number of strong variables in ϕ'. For

instance, the formulas from (4) all have strong quantifier rank 3 and, thus, the triangle packing problem can be described by a family of constant (normal) quantifier rank. Applying Chen et al.'s characterization yields membership in para-AC^0.

As a more complex example, let us sketch a "purely syntactic" proof of the result [5,6] that the embedding problem for graphs H of tree depth at most d lies in para-AC^0 for each d. Once more, we construct a family (ϕ_H) of formulas, one for each to-be-embedded graph H, of constant strong quantifier rank that describes the problem. For a graph $H = \big(V(H), E(H)\big)$ to have tree depth d means that there is a rooted tree T of depth d such that $E(H)$ is contained in the edges of T's transitive closure. Let c_1 be the root of T and let $\text{children}(c)$ be the (possibly empty) set of children of c in T. Then the following formula of strong quantifier rank d describes that H can be embedded into a graph structure $\mathcal{G} = (V, E^{\mathcal{G}})$ (note that in the formula the E in "En_in_j" refers to the edge relation $E^{\mathcal{G}}$ of $\mathcal{G}$ while in "$(c_i, c_j) \in E(H)$" it refers to the edge set $E(H)$ of the fixed parameter H):

$$\begin{aligned}\exists \dot{x}_1 \cdots \exists \dot{x}_{|V(H)|} \big(\textstyle\bigwedge_{i\neq j} \dot{x}_i \neq \dot{x}_j \wedge \exists n_1 (n_1 = \dot{x}_{c_1} \wedge \textstyle\bigwedge_{c_2 \in \text{children}(c_1)} \exists n_2 (n_2 = \dot{x}_{c_2} \wedge \\ \textstyle\bigwedge_{c_3 \in \text{children}(c_2)} \exists n_3 (n_3 = \dot{x}_{c_3} \wedge \textstyle\bigwedge_{c_4 \in \text{children}(c_3)} \exists n_4 (n_4 = \dot{x}_{c_4} \wedge \dots \\ \textstyle\bigwedge_{c_d \in \text{children}(c_{d-1})} \exists n_d (n_d = \dot{x}_{c_d} \wedge \textstyle\bigwedge_{i,j \in \{1,\dots,d\}:(c_i,c_j)\in E(H)} En_in_j) \dots)))))\big).\end{aligned}$$

Our Contributions III: Slicewise Descriptions and Variable Set Sizes.

Our third contribution is a new result in the same vein as the already repeatedly mentioned result of Chen et al. [4], stated as Fact 1 in our paper: Our new Theorem 1 states that a parameterized problem can be described slicewise by a family $(\phi_k)_{k\in\mathbb{N}}$ of arithmetic first-order formulas that all use only a bounded number of variables if and only if the problem lies in para-AC$^{0\uparrow}$—a class that has been encountered repeatedly in the literature [5,7–9], but for which no characterization was known. It contains all parameterized problems that can be decided by AC-circuits whose depth depends only on the parameter and whose size is of the form $f(k) \cdot n^c$.

As an example, consider the problem of deciding whether a graph contains a path of length k (no vertex may be visited twice). It can be described by the family (δ_k) of formulas with (for odd k) δ_k equal to

$$\begin{aligned}\exists s \exists t\, \exists x (Esx \wedge \exists \dot{x}_1 (\dot{x}_1 = x \wedge \\ \exists y (Exy \wedge \exists \dot{x}_2 (\dot{x}_2 = y \wedge \\ \exists x (Eyx \wedge \exists \dot{x}_3 (\dot{x}_3 = x \wedge \\ \exists y (Exy \wedge \exists \dot{x}_4 (\dot{x}_4 = y \wedge \\ \cdots \wedge \exists x (Eyx \wedge x = t \wedge \exists \dot{x}_k (\dot{x}_k = x \wedge \textstyle\bigwedge_{i\neq j} \dot{x}_i \neq \dot{x}_j) \dots))))))))).\end{aligned} \tag{5}$$

Note that the strong quantifier rank of δ_k is $k+2$ and, thus, depends on k. However, there are only four strong variables, namely s, t, x, and y. By Theorem 3 we see that the above formulas are equivalent to a family of formulas with a bounded number of variables and by Theorem 1 we see that p_k-LONG-PATH $\in$ para-AC$^{0\uparrow} \subseteq$ FPT. These ideas also generalize easily and we give a purely syntactic proof of the seminal result from the original color coding paper [3] that the embedding problem for graphs of bounded tree width lies in FPT. The core observation—which unifies the results for tree width and depth—is that for each graph with a given tree decomposition, the embedding problem can be described by a formula whose strong nesting structure mirrors the tree structure and whose strong variables mirror the bag contents.

Table 1 summarizes, for the problems discussed in this paper, how they can be described using formulas in such a way that membership in para-AC^0 and para-AC$^{0\uparrow}$ follows. All memberships were previously known, the contribution of this paper is that we prove the memberships by presenting families of first-order formulas that describe them, such that that strong quantifier rank (strong-qr) or the number of strong variables (strong-vars) is parameter-independent.

Table 1. Membership of the problems discussed in this paper in the two classes para-AC^0 and para-$AC^{0\uparrow}$ and the formulas whose syntactic structures prove this. Except for clique problem, formulas exist where at least the number of strong variables is independent of the parameter—but it is not always the most "natural" formulas that have this desirable property, see the vertex cover problem.

p_k-MATCHING $\in$ para-AC^0 Problem: Does a graph have a size-k matching? Formulas: $\exists \dot{x}_1 \cdots \exists \dot{x}_{2k} \big(\bigwedge_{i \neq j} \dot{x}_i \neq \dot{x}_j \wedge \bigwedge_{i=1}^{k} E\dot{x}_{2i-1}\dot{x}_{2i} \big)$, see Equation (1) Descriptive complexity: strong-qr $= 0$, strong-vars $= 0$
p_k-TRIANGLE-PACKING $\in$ para-AC^0 Problem: Does a graph contain k vertex-disjoint triangles? Formulas: $\exists \dot{x}_1 \cdots \exists \dot{x}_{3k} \big(\bigwedge_{i \neq j} \dot{x}_i \neq \dot{x}_j \wedge \bigwedge_{j=1}^{k} \exists x \exists y \exists z \, ($ $\dot{x}_{3j-2} = x \wedge \dot{x}_{3j-1} = y \wedge \dot{x}_{3j} = z \wedge Exy \wedge Exz \wedge Eyz) \big)$, see Equation (2) Descriptive complexity: strong-qr $= 3$, strong-vars $= 3$
p_k-H-PACKING $\in$ para-AC^0 for a fixed graph $H = (V(H), E(H))$ (instead of triangles) Problem: Does a graph contain k vertex-disjoint induced copies of H? Formulas: $\exists \dot{x}_1 \cdots \exists \dot{x}_{k\lvert V(H)\rvert} \big(\bigwedge_{i \neq j} \dot{x}_i \neq \dot{x}_j \wedge \bigwedge_{j=1}^{k} \exists y_1 \cdots \exists y_{\lvert V(H)\rvert} \, ($ $\bigwedge_{i=1}^{\lvert V(H)\rvert} y_i = \dot{x}_{(j-1)k+i} \wedge \bigwedge_{(p,q) \in E(H)} Ey_p y_q \wedge \bigwedge_{(p,q) \notin E(H)} \neg Ey_p y_q) \big)$ Descriptive complexity: strong-qr $= \lvert V(H)\rvert$, strong-vars $= \lvert V(H)\rvert$
p_k-LONG-PATH $\in$ para-$AC^{0\uparrow}$ Problem: Does a graph contain a path of length k? Formulas: $\exists s \exists t \exists x (Esx \wedge \exists \dot{x}_1 (\dot{x}_1 = x \wedge \exists y (Exy \wedge \ldots (\dot{x}_k = x \wedge \bigwedge_{i \neq j} \dot{x}_i \neq \dot{x}_j) \ldots)))$ Descriptive complexity: strong-qr $= k + 2$, strong-vars $= 4$, see Equation (5)
p_k-VERTEX-COVER $\in$ para-AC^0 Problem: Does a graph have a size-k vertex cover? Formulas 1: $\exists x_1 \cdots \exists x_k \forall u \forall v \big(Euv \to \bigvee_i (x_i = u \vee x_i = v) \big)$ Descriptive complexity: strong-qr $= k + 2$, strong-vars $= k + 2$ Formulas 2: Describe the Buss kernelization, see Theorem 5 Descriptive complexity: strong-qr $= O(1)$, strong-vars $= O(1)$
$\mathrm{p}_{k,d}$-HITTING-SET $\in$ para-AC^0 Problem: Does a hypergraph with maximum edge size d a size-k hitting set? Formulas: Describe a kernel based on pseudo-sunflowers, see Theorem 6 Descriptive complexity: strong-qr $= O(1)$, strong-vars $= O(1)$
$\mathrm{p}_{\psi,\delta}$-MC(FO) $\in$ para-$AC^{0\uparrow}$ Problem: Is ψ a model of a graph with maximum degree δ? Formulas: Describe the disjointness of balls in the formula resulting from Gaifman's Theorem using weak variables, see Theorem 7 Descriptive complexity: strong-qr $= O(\text{locality-rank}(\psi))$, strong-vars $= O(1)$
p_H-$\mathrm{EMB}_{\mathrm{td}(H) \leq c}$ $\in$ para-AC^0 and p_H-$\mathrm{EMB}_{\mathrm{tw}(H) \leq c}$ $\in$ para-$AC^{0\uparrow}$ Problem: Can H, of tree depth or width c, be embedded into a graph G? Formulas: Bind the vertices of H's image to weak variables and use a formula whose syntactic structure exactly mimics an optimal tree decomposition of H, see Theorem 8 Descriptive complexity: strong-qr $= O(\mathrm{td}(H))$, strong-vars $= O(\mathrm{tw}(H))$
p_k-CLIQUE $\in$ W[1] Problem: Does a graph contain a k-clique? Formulas 1: $\exists x_1 \cdots \exists x_k \big(\bigwedge_{i \neq j} x_i \neq x_j \wedge \bigwedge_{i \neq j} Ex_i x_j \big)$, see Equation (3) Formulas 2: $\exists x_1 \cdots \exists x_k \forall u \forall v \big(\bigvee_{i \neq j} (x_i = u \wedge x_j = v) \to Euv \big)$, see Equation (19) Descriptive complexity: strong-qr $= k$, strong-vars $= k$ in both cases

1.1. Related Work

Flum and Grohe [10] were the first to give characterizations of FPT and of many subclasses in terms of the syntactic properties of formulas describing their members. Unfortunately, these syntactic properties do not hold for the descriptions of parameterized problems found in the literature. For instance, they show that FPT contains exactly the problems that can be described by families of FO[LFP]-formulas of bounded quantifier rank—but actually describing problems like p_k-VERTEX-COVER in this way is more or less hopeless and yields little insights into the structure or complexity of the problem. We believe that it is no coincidence that no applications of these beautiful characterizations to concrete problems could be found in the literature—at least prior to very recent work by Chen and Flum [11], who study slicewise descriptions of problems on structures of bounded tree depth, and the already cited article of Chen et al. [4], who do present a family of formulas that describe the vertex cover problem. This family internally uses the color coding technique and is thus closely related to our results. The crucial difference is, however, that we identify syntactic properties of logical formulas that imply that the color coding technique can be applied. It then suffices to find a family describing a given problem that meets the syntactic properties to establish the complexity of the problem: there is no need to actually construct the color-coding-based formulas—indeed, there is not even a need to understand how color coding works in order to decide whether a quantifier is weak or strong.

A different approach towards proving that a problem lies in FPT purely because of the syntactic nature of logic-based problem descriptions comes from the context of optimization problems. Cai and Chen [12] have shown that all problems in the syntactically-defined classes MAXSNP, due to [13], and MIN$\mathrm{F}^+\Pi_1$, due to [14], are fixed-parameter tractable (but not necessarily in the small classes like para-AC^0 and para-$\mathrm{AC}^{0\uparrow}$ studied in the present paper).

1.2. Organization of this Paper

In Section 2 we first review some of the existing work on the descriptive complexity of parameterized problems. We add to this work in the form of the mentioned characterization of the class para-$\mathrm{AC}^{0\uparrow}$ in terms of a bounded number of variables. Our main technical results are then proved in Section 3, where we establish and prove the syntactic properties that formulas must have in order for the color coding method to be applicable. In Section 4 we then apply the findings and show how membership of different natural problems in para-AC^0 and para-$\mathrm{AC}^{0\uparrow}$ (and, thus, in FPT) can be derived entirely from the syntactic structure of the formulas describing them.

2. Describing Parameterized Problems

A happy marriage of parameterized complexity and descriptive complexity was first presented in [10] by Flum and Grohe. We first review their most important definitions and then prove a new characterization, namely of the class para-$\mathrm{AC}^{0\uparrow}$ that contains all problems decidable by AC-circuits of parameter-dependent depth and "FPT-like" size.

2.1. Logical Terminology

We only consider first-order logic and use standard notations following for instance [10], with the perhaps only deviations being that we write relational atoms briefly as Exy instead of $E(x, y)$ and that the literal $x \neq y$ is an abbreviation for $\neg\, x = y$ (recall that a literal is an atom or a negated atom). Signatures, typically denoted τ, are always finite and may only contain relation symbols and constant symbols—with one exception: The special unary function symbol SUCC may also be present in a signature. Let us write SUCC^k for the k-fold application of SUCC, so $\mathrm{SUCC}^3(x)$ is short for $\mathrm{SUCC}(\mathrm{SUCC}(\mathrm{SUCC}(x)))$. It allows us to specify any fixed non-negative integer without having to use additional variables. An alternative is to dynamically add constant symbols for numbers to signatures as done

in [4], but we believe that following [10] and adding the successor function gives a leaner formal framework. Let arity(τ) be the maximum arity of relation symbols in τ.

We denote by $\text{STRUC}[\tau]$ the class of all τ-structures and by $|\mathcal{A}|$ the universe of $\mathcal{A}$. As is often the case in descriptive complexity theory, we only consider ordered structures in which the ternary predicates ADD and MULT are available and have their natural meaning. Formally, we say τ is arithmetic if it contains all of the predicates $<$, ADD, MULT, the function symbol SUCC, and the constant symbol 0 (it is included for convenience only). In this case, $\text{STRUC}[\tau]$ contains only those $\mathcal{A}$ for which $<^{\mathcal{A}}$ is a linear ordering of $|\mathcal{A}|$ and the other operations have their natural meaning relative to $<^{\mathcal{A}}$ (with the successor of the maximum element of the universe being itself and with 0 being the minimum with respect to $<^{\mathcal{A}}$). We write $\phi \in \text{FO}[+,\times]$ when ϕ is a τ-formula for an arithmetic τ.

A τ-problem is a set $Q \subseteq \text{STRUC}[\tau]$ closed under isomorphisms. A τ-formula ϕ describes a τ-problem Q if $Q = \{\mathcal{A} \in \text{STRUC}[\tau] \mid \mathcal{A} \models \phi\}$ and it describes Q eventually if ϕ describes a set Q' that differs from Q only on structures of a certain maximum size.

Lemma 1. *For each $\phi \in \text{FO}[+,\times]$ that describes a τ-problem Q eventually, there are quantifier-free formulas α and β such that $(\alpha \wedge \phi) \vee \beta$ describes Q.*

Proof. The statement of the lemma would be quite simple if we did not require α and β to be quantifier-free: Without this requirement, all we need to do is to use α and β to fix ϕ on the finitely many (up to isomorphisms) structures on which ϕ errs by "hard-wiring" which of these structures are elements of Q and which are not. However, the natural way to do this "hard-wiring" of size-m structures is to use m quantifiers to bind all elements of the universe. This is exactly what we do not wish to do. Rather, we use the successor function to refer to the elements of the universe without using any quantifiers.

In detail, let m be a number such that for all $\mathcal{A} \in \text{STRUC}[\tau]$ with $\|\mathcal{A}\| \geq m$ (that is, the size $\|\mathcal{A}\|$ of the universe $|\mathcal{A}|$ is at least m) we have $\mathcal{A} \models \phi$ if and only if $\mathcal{A} \in Q$. We set α to $\text{UNIVERSE}^{\geq m}$, a shorthand for $\text{SUCC}^{m-1}(0) \neq \text{SUCC}^{m}(0)$, which is true only for universes of size at least m. We define β so that it is true exactly for all τ-structures $\mathcal{A} \in Q$ of size at most m (for simplicity we assume that E^2 is the only relation symbol in τ):

$$\beta = \bigwedge_{s=1}^{m} \Big((\text{UNIVERSE}^{\geq s} \wedge \neg\, \text{UNIVERSE}^{\geq s+1}) \rightarrow \bigvee_{\mathcal{A} \in Q, |\mathcal{A}| = \{0,\ldots,s-1\}} \Big(\bigwedge_{u,v \in |\mathcal{A}| : (u,v) \in E^{\mathcal{S}}} E(\text{SUCC}^u(0), \text{SUCC}^v(0)) \wedge \bigwedge_{u,v \in |\mathcal{A}| : (u,v) \notin E^{\mathcal{S}}} \neg E(\text{SUCC}^u(0), \text{SUCC}^v(0)) \Big) \Big).$$

The first line of β checks whether the universe of the current structure (the structure for which we would like to know whether it is a model of $(\alpha \wedge \phi) \vee \beta$) has size s for some $s \leq m$ and, if so, checks that there is some $\mathcal{A} \in Q$ of size exactly s so that the edges of $\mathcal{A}$ are present in the current structure (second line) and that the edges not in $\mathcal{A}$ are also not present in the current structure (last line). In particular, if the current structure has size at most m, it will be a model of β if and only if it is isomorphic to some $\mathcal{A} \in Q$ and, thus, if it is an element of Q. □

We write $\text{qr}(\phi)$ for the quantifier rank of a formula and $\text{bound}(\phi)$ for the set of its bound variables. For instance, for $\phi = (\exists x \exists y(Exz)) \vee \forall y(Px)$ we have $\text{qr}(\phi) = 2$, since the maximum nesting is caused by the two nested existential quantifiers, and $\text{bound}(\phi) = \{x, y\}$.

Let us say that ϕ is in negation normal form if negations are applied only to atomic formulas.

2.2. Describing Parameterized Problems

When switching from classical complexity theory to descriptive complexity theory, the basic change is that "words" get replaced by "finite structures." The same idea works for parameterized complexity theory and, following Flum and Grohe [10], let us define

parameterized problems as subsets $Q \subseteq \text{STRUC}[\tau] \times \mathbb{N}$ where Q is closed under isomorphisms. In a pair $(\mathcal{A}, k) \in \text{STRUC}[\tau] \times \mathbb{N}$ the number k is, of course, the parameter value of the pair. Flum and Grohe now propose to describe such problems slicewise using formulas. Since this will be the only way in which we describe problems, we will drop the "slicewise" in the phrasings and just say that a family $(\phi_k)_{k\in\mathbb{N}}$ of formulas describes a problem $Q \subseteq \text{STRUC}[\tau] \times \mathbb{N}$ if for all $(\mathcal{A}, k) \in \text{STRUC}[\tau] \times \mathbb{N}$ we have $(\mathcal{A}, k) \in Q$ if and only if $\mathcal{A} \models \phi_k$. (In this paper, we ignore the question of whether the mappings $k \mapsto \phi_k$ should be required to be computable. While this will be the case for all of our examples and constructions, it is not important for our formalism and results.)

For a class Φ of families $(\phi_k)_{k\in\mathbb{N}}$, let us write $\text{X}\Phi$ for the class of all parameterized problems that are described by the members of Φ (we chose "X" to represent a "slicewise" description, which seems to be in good keeping with the usual use of X in other classes such as XP or XL). For instance, the mentioned characterization of FPT in logical terms by Flum and Grohe can be written as

$$\text{FPT} = \text{X}\big\{(\phi_k)_{k\in\mathbb{N}} \mid \phi_k \in \text{FO}[\text{LFP}], \max_k \text{qr}(\phi_k) < \infty\big\}.$$

We remark that instead of describing parameterized problems using families, a more standard and at the same time more flexible way is to use reductions to model checking problems. Clearly, if a family $(\phi_k)_{k\in\mathbb{N}}$ of $\mathcal{L}$-formulas describes $Q \subseteq \text{STRUC}[\tau] \times \mathbb{N}$, then there is a very simple parameterized reduction from Q to the model checking problem $\text{p}_\phi\text{-MC}(\mathcal{L})$, where the input is a pair $(\mathcal{A}, \text{num}(\phi))$ and the question is whether both $\mathcal{A} \models \phi$ and $\phi \in \mathcal{L}$ hold. (The function num encodes mathematical objects like ϕ or later tuples like (ϕ, δ) as unique natural numbers.) The reduction simply maps a pair $(\mathcal{A}, k)$ to $(\mathcal{A}, \text{num}(\phi_k))$. Even more interestingly, without going into any of the technical details, it is also not hard to see that as long as a reduction is sufficiently simple, the reverse implication holds, that is, we can replace a reduction to the model checking problem by a family of formulas that describe the problem. We can, thus, use whatever formalism seems more appropriate for the task at hand and—as we hope that this paper shows—it is sometimes quite natural to write down a family that describes a problem.

2.3. Parameterized Circuits

For our descriptive setting, we need to slightly adapt the definition of the circuit classes para-AC^0 and para-$\text{AC}^{0\uparrow}$ from [5,7]: Let us say that a problem $Q \subseteq \text{STRUC}[\tau] \times \mathbb{N}$ is in para-AC^0, if there is a family $(C_{n,k})_{n,k\in\mathbb{N}}$ of AC-circuits (Boolean circuits with unbounded fan-in) such that for all $(\mathcal{A}, k) \in \text{STRUC}[\tau] \times \mathbb{N}$ we have

1. $(\mathcal{A}, k) \in Q$ if and only if $C_{|x|,k}(x) = 1$ where x is a binary encoding of $\mathcal{A}$ and $|x|$ is the length of the encoding,
2. the size of each $C_{n,k}$ is at most $f(k) \cdot n^c$ for some function f and some constant c that is independent of both n and k,
3. the depth of each $C_{n,k}$ is bounded by some constant that is again independent of both n and k, and
4. the circuit family satisfies a DLOGTIME-uniformity condition. (Since the complex questions around circuit uniformity are not of special importance for the present paper, we keep its discussion short and just remark that DLOGTIME-uniformity roughly means that arithmetic formulas suffice to describe the circuits.)

The class para-$\text{AC}^{0\uparrow}$ is defined the same way, but the depth may be $g(k)$ for some g instead of only $O(1)$. The following fact and theorem show how these two circuit classes are closely related to descriptions of parameterized problems using formulas:

Fact 1 ([4]). $\text{para-AC}^0 = \text{X}\big\{(\phi_k)_{k\in\mathbb{N}} \mid \phi_k \in \text{FO}[+, \times], \max_k \text{qr}(\phi_k) < \infty\big\}$.

Theorem 1. $\text{para-AC}^{0\uparrow} = \text{X}\big\{(\phi_k)_{k\in\mathbb{N}} \mid \phi_k \in \text{FO}[+, \times], \max_k |\text{bound}(\phi_k)| < \infty\big\}$.

Proof. The basic idea behind the proof is quite "old." We wish to establish links between circuit depth and size and the number of variables used in a formula—and such links are well-known, see for instance [15]: The quantifier rank of a first-order formula naturally corresponds to the depth of a circuit that solves the model checking problem for the formula. The number of variables corresponds to the exponent of the polynomial that bounds the size of the circuit (the paper [16] is actually entitled "$\mathrm{DSPACE}[n^k] = \mathrm{VAR}[k+1]$"). One thing that is usually not of interest (because only one formula is usually considered) is the fact that the length of the formula is linked multiplicatively to the size of the circuit.

In detail, suppose we are given a problem $Q \subseteq \mathrm{STRUC}[\tau] \times \mathbb{N}$ with $Q \in \text{para-AC}^{0\uparrow}$ via a circuit family $(C_{n,k})_{n,k\in\mathbb{N}}$ of depth $g(k)$ and size $f(k)n^c$. For a fixed k, we now need to construct a formula ϕ_k that correctly decides the k-th slice. In other words, we need an $\mathrm{FO}[+,\times]$-formula ϕ_k whose finite models are exactly those on which the family $(C_{n,k})_{n\in\mathbb{N}}$ (note that k no longer indexes the family) evaluates to 1 (when the models are encoded as bitstrings). It is well-known how such a formula can be constructed, see for instance [15], we just need a closer look at how the quantifier rank and number of variables relate to the circuit depth and size.

The basic idea behind the formula ϕ_k is the following: The circuit has $f(k)n^c$ gates and we can "address" these gates using c variables (which gives us n^c possibilities) plus a number $i \in \{1,\dots,f(k)\}$ (which gives us $f(k)\cdot n^c$ possibilities). Since for fixed k the number $f(k)$ is also fixed, it is permissible that the formula ϕ_k contains $f(k)$ copies of some subformula, where each subformula handles another value of i. The basic idea is then to start with formulas ψ_i^0 for $i \in \{1,\dots,f(k)\}$, each of which has c free variables, so that $\psi_i^0(x_1,\dots,x_c)$ is true exactly if the tuple $(x_1,\dots,x_c,i)$ represents an input gate set to 1. At this point, the uniformity condition basically tells us that arithmetic formulas suffice to express this and that they all have a fixed quantifier rank. Next, we construct formulas $\psi_i^1(x_1,\dots,x_c)$ that are true if $(x_1,\dots,x_c,i)$ addresses a gate for which the input values are all already computed by the ψ_j^0 and that evaluates to 1. Next, formulas ψ_i^2 are constructed, but now, we can reuse the variables used in the ψ_j^0. In this way, we finally build formulas $\psi_i^{g(k)}$ and apply it to the "address" of the output gate. All told, we get a formula whose quantifier rank is $c\cdot g(k)+O(1)$ and in which at most $2c+O(1)$ variables are used (note that the size of the formula depends on $f(k)$). Clearly, this means that the family $(\phi_k)_{k\in\mathbb{N}}$ created in this way does, indeed, only use a bounded number of variables (namely $O(c)$ many) and decides Q.

For the other direction, suppose $(\phi_k)_{k\in\mathbb{N}}$ describes Q and that all ϕ_k contain at most v variables (since they contain no free variables, this is same as the number of bound variables). Clearly, we may assume that the formulas ϕ_k are in negation normal form. We may also assume that they are "flat," by which we mean that they contain no subformulas of the form $(\alpha\vee\beta)\wedge\gamma$ or $\alpha\wedge(\beta\vee\gamma)$: Using the distributive laws of propositional logic, any first-order formula can be turned into an equivalent flat formula with the same number of variables and the same quantifier rank (one can loosely think of this as locally transforming the formula into disjunctive normal form, but "not past quantifiers"). Lastly, we may assume that the SUCC function symbol is only used in atoms of the form $x = \mathrm{SUCC}^s(0)$ for some variable x and some number s: We can replace for instance $E\,\mathrm{SUCC}^6(x)\,\mathrm{SUCC}^3(y)$ by the equivalent formula $\exists x'\exists x''\exists y'\exists y''(x' = \mathrm{SUCC}^6(0)\wedge \mathrm{ADD}\,xx'x''\wedge y' = \mathrm{SUCC}^3(0)\wedge \mathrm{ADD}\,yy'y''\wedge Ex''y'')$ without raising the number of variables and the quantifier rank by more than 4 (or, in general, by more than the constant $2\cdot\mathrm{arity}(\tau)$).

As before, it is now known that for each ϕ_k there is a family $(C_{n,k})_{n\in\mathbb{N}}$ that evaluates to 1 exactly on the (encoded) models of ϕ_k. These circuits are constructed as follows: While ϕ_k has no free variables, a subformula ψ of ϕ_k can have up to v free variables. For each such subformula, the circuits use n^v gates to keep track of all assignments to these v variables that make the subformula true. Clearly, this is relatively easy to achieve for literals in a constant number of layers, including literals of the form $x = \mathrm{SUCC}^s(0)$ since s is a fixed number depending only on k. Next, if a formula is of the form $\bigwedge_i \alpha_i$ and for some

assignment we have one gate for each α_i that tells us whether it is true, we can feed all these wires into one $\wedge$-gate. We can take care of a formula of the form $\bigvee_i \alpha_i$ in the same way—and note that in a flat formula there will be at most one alternation from $\bigwedge$ to $\bigvee$ before we encounter a quantifier. Now, for subformulas of the form $\exists x\, \phi$, the correct values for the n^{v-1} gates can be obtained by a big $\vee$-gate attached to the outputs from the gates for ϕ. Similarly, $\forall x\, \phi$ can be handled using a big $\wedge$-gate.

Based on these observations, it is now possible to build a circuit of size $|\phi_k| n^v$ and depth $O(\mathrm{qr}(\phi_k))$. In particular, the resulting overall circuit family has a depth that depends only on the parameter (since the quantifier rank can be at most $|\phi_k|$, which depends only on k) and has a size of at most $f(k) \cdot n^c$ for $f(k) = |\phi_k|$. It can also be shown that the necessary uniformity conditions are satisfied.

We remark that the above proof also implies Fact 1, namely for $g(k) = O(1)$ for the first direction and for $\mathrm{qr}(\phi_k) = O(1)$ for the second direction. □

2.4. Bounded Rank Reductions

To complete the formal framework for describing parameterized problems, we need a notion of parameterized reductions that is very weak to ensure that the smallest class we study, para-AC^0, is closed under them. Such a reduction is used in the literature [5], boringly named para-AC^0-reduction, but both its definition as well as the definition of other kinds of parameterized reductions found in the literature do not fit well with our logical framework: The reductions are defined in terms of machines or circuits that get as input a string that explicitly or implicitly contains the parameter k and output a new problem instance that once more explicitly or implicitly contains a new parameter value k'.

In contrast, in our setting the inputs and outputs must be logical structures that we wish to define in terms of formulas. Furthermore, "outputting a parameter value" is difficult in our formal framework since parameter values are not elements of the universe, but indices of the formulas. All of these problems can be circumvented, see for instance ([11], Definition 5.3), but we believe it gives a cleaner formalism to give a new "purely logical" definition of reductions between parameterized problems. We will not prove this, but remark that the power of this reduction is the same as that of para-AC^0-reductions.

Definition 1. *Let τ and τ' be signatures and let $Q \subseteq \mathrm{STRUC}[\tau] \times \mathbb{N}$ and $Q' \subseteq \mathrm{STRUC}[\tau'] \times \mathbb{N}$ be two problems. A bounded rank reduction from Q to Q', written $Q \leq_{\mathrm{br}} Q'$, is a pair of families $(f_k)_{k\in\mathbb{N}}$ and $(\iota_{k,k'})_{k,k'\in\mathbb{N}}$ where*

- *each f_k is a first-order query from τ-structures to τ'-structures and*
- *each $\iota_{k,k'}$ is a τ-formula*

such that

1. *for each $(\mathcal{A}, k) \in \mathrm{STRUC}[\tau] \times \mathbb{N}$ there is exactly one $k' \in \mathbb{N}$, denoted by $\iota_k(\mathcal{A})$ in the following, such that $\mathcal{A} \models \iota_{k,k'}$,*
2. *there is a mapping $\iota^* \colon \mathbb{N} \to \mathbb{N}$ such that for all $\mathcal{A} \in \mathrm{STRUC}[\tau]$ we have $\iota_k(\mathcal{A}) \leq \iota^*(k)$,*
3. *$(\mathcal{A}, k) \in Q$ if and only if $\big(f_k(\mathcal{A}), \iota_k(\mathcal{A})\big) \in Q'$, and*
4. *the quantifier rank of all $\iota_{k,k'}$ and of all formulas inside the f_k and of the widths of each f_k is bounded by a constant c.*

Let us briefly explain the ingredients of this definition: Each f_k maps all τ-structures $\mathcal{A}$ to τ'-structures $\mathcal{A}'$. The fact that we have one function for each parameter value allows us to make our mapping depend on the parameter. The job of the formulas $\iota_{k,k'}$ is solely to "compute" the new parameter value k', based not only on the original value k, but also on $\mathcal{A}$. If, as is the case in many reductions, the new parameter value k' just depends on k (typically, it even is k), we can just set $\iota_{k,k'}$ to a trivial tautology $\top$ and all other $\iota_{k,k''}$ to the contradiction $\bot$.

In the definition, we referred to first-order queries, which are a standard way of defining a logical τ'-structure in terms of a τ-structure in database theory and finite model theory (we remark that in general model theory "interpretation" is used instead of "query"

and that "transductions" from FPT-theory are queries with special semantic and syntactic properties). A detailed account of first-order queries can be found in [17], but here is the basic idea: Suppose we wish to map graphs ((E^2)-structures) to their underlying undirected graphs ((U^2)-structures, where U represent the underlying symmetric edge set). In this case, there is a simple formula $\phi_U(x,y)$ that tells us when Uxy holds in the new structure: $Exy \vee Eyx$. More importantly, if we have a formula ψ that internally uses Uxy to check whether there is an undirected edge in the mapped graph, we can easily turn this into a formula $\psi[f]$, where we replace all occurrences of Uxy by $\phi_U(x,y)$, that gives the same answer as ψ when fed the original graph. In other words, if a first-order query maps $\mathcal{A}$ to $\mathcal{A}'$ and we wish to check whether $\mathcal{A}' \models \psi$ holds, we can just as well check whether $\mathcal{A} \models \psi[f]$ holds. The just-described example of a first-order query did not change the universe, which is something we sometimes wish to do. This is achieved by allowing the width w of the query to be larger than 1. The effect is that the universe U gets replaced by U^w and, now, elements of this new universe can be described by tuples of variables of length w. We can also reduce the size of the universe using a formula $\phi_{\text{universe}}(x_1,\dots,x_w)$ that is true only for the tuples we wish to keep in the new structure's universe.

Lemma 2. *Let $Q \leq_{\text{br}} Q'$ via a bounded rank reduction given by $(f_k)_{k\in\mathbb{N}}$ and $(\iota_{k,k'})_{k,k'\in\mathbb{N}}$. Let $(\phi'_k)_{k\in\mathbb{N}}$ describe Q'. Then there is a family $(\phi_k)_{k\in\mathbb{N}}$ that describes Q with*

1. $\max_k \operatorname{qr}(\phi_k) = \max_k \operatorname{qr}(\phi'_k) + O(1)$ *and*
2. $\max_k |\operatorname{bound}(\phi_k)| = \max_k |\operatorname{bound}(\phi'_k)| + O(1)$.

In particular, para-AC^0 and para-$\mathrm{AC}^{0\uparrow}$ are closed under bounded rank reductions.

Proof. Set ϕ_k to $\bigwedge_{k'=1}^{i^*(k)}(\iota_{k,k'} \to \phi'_{k'}[f_k])$. By definition, we have $\mathcal{A} \models \phi_k$ if and only if $f_k(\mathcal{A}) \models \phi'_{k'}$ for the unique k' with $\mathcal{A} \models \iota_{k,k'}$. If we can argue that the substitutions do not increase the quantifier rank or number of variables by more than a constant, we get the claim. Unfortunately, there is a case where simple substitutions fail to preserve the quantifier rank, namely when a formula $\phi'_{k'}$ contains a large number of nested applications of the successor function. Suppose, for instance, $\phi'_{k'}$ is something like $\exists x \exists y(\mathrm{SUCC}^{1000}\, x = y)$. While this formula has quantifier rank 2 and uses only two variables, a simple substitution of each occurrence of the one thousand SUCC operators in $\phi'_{k'}$ by any nontrivial formula in f_k that describes the successor function will yield a quantifier rank of at least 1000.

The trick is to use the results from the next section: We can easily modify any formula so that all occurrences of the successor function are of the form $x = \mathrm{SUCC}^i\, 0$ for some number i. This means that we "only" need a way of identifying the ith element of the new universe using a bounded quantifier rank. However, assuming for simplicity a width of 1 and assuming that $\phi_{\text{universe}}(x)$ and $\phi_<(x,y)$ describe how f_k restricts and reorders the universe, respectively, the formula $\phi_{\text{universe}}(x) \wedge \exists^{=i-1} y(\phi_{\text{universe}}(y) \wedge \phi_<(y,x))$ is true exactly for the ith element of the universe. We will see in the next section that we can express the $\exists^{=i-1}y$ quantifier using a constant quantifier rank that is independent of i. □

3. Syntactic Properties Allowing Color Coding

The color coding technique [3] is a powerful method from parameterized complexity theory for "discovering small objects" in larger structures. Recall the example from the introduction: While finding k disjoint triangles in a graph is difficult in general, it is easy when the graph is colored with k colors and the objective is to find for each color one triangle having this color. The idea behind color coding is to reduce the (hard) uncolored version to the (easy) colored version by randomly coloring the graph and then "hoping" that the coloring assigns a different color to each triangle. Since the triangles are "small objects," the probability that they do, indeed, get different colors depends only on k. Even more importantly, Alon et al. noticed that we can derandomize the coloring procedure simply by coloring each vertex by its hash value with respect to a simple family of universal hash functions that only use addition and multiplication [3]. This idea is beautiful and

works surprisingly well in practice [18], but using the method inside proofs can be tricky: One needs to "keep the set sizes under control" (they must stay roughly logarithmic in size) and one needs to algorithmically "identify the small set based just on its random coloring," which especially for more complex proofs can lead to rather subtle arguments.

In the present section, we identify syntactic properties of formulas that guarantee that the color coding technique can be applied. The property is that the colors (the predicates C_i in the formulas) are not in the scope of a universal quantifier (this restriction is necessary, as the example of the formula describing 3-colorability shows).

As mentioned already in the introduction, the main "job" of the colors in proofs based on color coding is typically to ensure that vertices of a graph are different from other vertices. This leads us to the idea of focusing entirely on the notion of distinctness in the second half of this section. This time, there will be syntactic properties of existentially bounded first-order variables that will allow us to apply color coding to them.

3.1. Formulas with Color Predicates

In graph theory, a coloring of a graph can either refer to an arbitrary assignment that maps each vertex to a color or to such an assignment in which vertices connected by an edge must get different colors (sometimes called proper colorings). For our purposes, colorings need not be proper and are thus partitions of the vertex set into color classes. From the logical point of view, each color class can be represented by a unary predicate. A k-coloring of a τ-structure $\mathcal{A}$ is a structure $\mathcal{B}$ over the signature $\tau_{k\text{-colors}} = \tau \cup \{C_1^1, \dots, C_k^1\}$, where the C_i are fresh unary relation symbols, such that $\mathcal{A}$ is the τ-restriction of $\mathcal{B}$ and such that the sets $C_1^{\mathcal{B}}$ to $C_k^{\mathcal{B}}$ form a partition of the universe $|\mathcal{A}|$ of $\mathcal{A}$.

Let us now formulate and prove the first syntactic version of color coding. An example of a possible formula ϕ in the below theorem is $\bigwedge_{i=1}^{k} \exists x \exists y \exists z (Exy \wedge Eyz \wedge Exz \wedge C_i x \wedge C_i y \wedge C_i z)$, for which the theorem tells us that there is a formula ϕ' of constant quantifier rank that is true exactly when there are pairwise disjoint sets C_i that make ϕ true.

Theorem 2. *Let τ be an arithmetic signature and let k be a number. For each first-order $\tau_{k\text{-colors}}$-sentence ϕ in negation normal form in which no C_i is inside a universal scope, there is a τ-sentence ϕ' such that:*

1. *For all $\mathcal{A} \in \text{STRUC}[\tau]$ we have $\mathcal{A} \models \phi'$ if and only if there is a k-coloring $\mathcal{B}$ of $\mathcal{A}$ with $\mathcal{B} \models \phi$.*
2. $\text{qr}(\phi') = \text{qr}(\phi) + O(1)$.
3. $|\text{bound}(\phi')| = |\text{bound}(\phi)| + O(1)$.

(Let us clarify that $O(1)$ represents a global constant that is independent of τ and k.)

Proof. Let τ, k, and ϕ be given as stated in the theorem. If necessary, we modify ϕ to ensure that there is no literal of the form $\neg C_i x_j$, by replacing each such literal by the equivalent $\bigvee_{l \neq i} C_l x_j$. After this transformation, the C_i in ϕ are neither in the scope of universal quantifiers nor of negations—and this is also true for all subformulas α of ϕ. We will now show by structural induction that all these subformulas (and, hence, also ϕ) have two semantic properties, which we call the monotonicity property and the small witness property (with respect to the C_i). Afterwards, we will show that these two properties allow us to apply the color coding technique.

Establishing the monotonicity and small witness properties. Some notations will be useful: Given a τ-structure $\mathcal{A}$ with universe A and given sets $A_i \subseteq A$ for $i \in \{1, \dots, k\}$, let us write $\mathcal{A} \models \phi(A_1, \dots, A_k)$ to indicate that $\mathcal{B}$ is a model of ϕ where $\mathcal{B}$ is the $\tau_{k\text{-colors}}$-structure with universe A in which all symbols from τ are interpreted as in $\mathcal{A}$ and in which the symbol C_i is interpreted as A_i, that is, $C_i^{\mathcal{B}} = A_i$. Subformulas γ of ϕ may have free variables and suppose that x_1 to x_m are the free variables in γ and let $a_i \in A$ for $i \in \{1, \dots, m\}$. We write $\mathcal{A} \models \gamma(A_1, \dots, A_k, a_1, \dots, a_m)$ to indicate that γ holds in the just-described structure $\mathcal{B}$ when each x_i is interpreted as a_i.

Definition 2. *Let γ be a $\tau_{k\text{-colors}}$-formula with free variables x_1 to x_m. We say that γ has the monotonicity and the small witness properties with respect to the C_i if for all τ-structures $\mathcal{A}$ with universe $A = |\mathcal{A}|$ and all values $a_1, \ldots, a_m \in A$ the following holds:*

1. *Monotonicity property: Let $A_1, \ldots, A_k \subseteq A$ and $B_1, \ldots, B_k \subseteq A$ be sets with $A_i \subseteq B_i$ for all $i \in \{1, \ldots, k\}$. Then $\mathcal{A} \models \gamma(A_1, \ldots, A_k, a_1, \ldots, a_m)$ implies $\mathcal{A} \models \gamma(B_1, \ldots, B_k, a_1, \ldots, a_m)$.*
2. *Small witness property: If there are sets $B_1, \ldots, B_k \subseteq A$ such that we have $\mathcal{A} \models \gamma(B_1, \ldots, B_k, a_1, \ldots, a_m)$, then there are sets $A_i \subseteq B_i$ whose sizes $|A_i|$ depend only on γ for $i \in \{1, \ldots, k\}$, such that $\mathcal{A} \models \gamma(A_1, \ldots, A_k, a_1, \ldots, a_m)$.*

We now show that ϕ has these two properties. For monotonicity, just note that the C_i are not in the scope of any negation and, thus, if some A_i make ϕ true, so will all supersets B_i of the A_i.

To see that the small witness property holds, we argue by structural induction: If ϕ is any formula that does not involve any C_i, then ϕ is true or false independently of the B_i and, in particular, if it is true at all, it is also true for $A_i = \emptyset$ for $i \in \{1, \ldots, k\}$. If ϕ is the atomic formula $C_i x_j$, then setting $A_i = \{a_j\}$ and $A_{i'} = \emptyset$ for $i' \neq i$ makes the formula true.

If $\phi = \alpha \wedge \beta$, then α and β have the small witness property by the induction hypothesis. Let $B_1, \ldots, B_k \subseteq A$ make ϕ hold in $\mathcal{A}$. Then they also make both α and β hold in $\mathcal{A}$. Let $A_1^\alpha, \ldots, A_k^\alpha \subseteq A$ with $A_i^\alpha \subseteq B_i$ be the witnesses for α and let $A_1^\beta, \ldots, A_k^\beta \subseteq A$ be the witnesses for β. Then by the monotonicity property, $A_1^\alpha \cup A_1^\beta, \ldots, A_k^\alpha \cup A_k^\beta$ makes both α and β true, that is

$$\mathcal{A} \models \alpha(A_1^\alpha \cup A_1^\beta, \ldots, A_k^\alpha \cup A_k^\beta, a_1, \ldots, a_m)$$

and the same holds for β. Note that $A_i^\alpha \cup A_i^\beta \subseteq B_i$ still holds and that they have sizes depending only on α and β and thereby on ϕ.

For $\phi = \alpha \vee \beta$ we can argue in exactly the same way as for the logical and.

The last case for the structural induction is $\phi = \exists x_m(\alpha)$. Consider $B_1, \ldots, B_k \subseteq A$ that make ϕ true. Then there is a value $a_m \in A$ such that $\mathcal{A} \models \alpha(B_1, \ldots, B_k, a_1, \ldots, a_m)$. Now, since α has the small witness property by the induction hypothesis, we get $A_i \subseteq B_i$ of size depending on α for which we also have $\mathcal{A} \models \alpha(A_1, \ldots, A_k, a_1, \ldots, a_m)$. Then, by the definition of existential quantifiers, these A_i also witness $\mathcal{A} \models \exists x_m \phi(A_1, \ldots, A_k, a_1, \ldots, a_{m-1})$. (Observe that this is the point where the argument would not work for a universal quantifier: Here, for each possible value of a_m we might have a different set of A_i's as witnesses and their union would then no longer have small size.)

Applying color coding: our next step in the proof is to use color coding to produce the partition. First, let us recall the basic lemma on universal hash functions formulated below in a way equivalent to ([2], p. 347):

Lemma 3. *There is an $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$ and all subsets $X \subseteq \{0, \ldots, n-1\}$ there exist a prime $p < |X|^2 \log_2 n$ and a number $q < p$ such that the function $h_{p,q}(m) = (q \cdot m \bmod p) \bmod |X|^2$ is injective on X.*

As has already been observed by Chen et al. [4], if we set $k = |X|$ we can easily express the computation underlying $h_{p,q} \colon \{0, \ldots, n-1\} \to \{0, \ldots, k^2 - 1\}$ using a fixed FO$[+, \times]$-formula $\rho(k, p, q, x, y)$. That is, if we encode the numbers $k, p, q, x, y \in \{0, \ldots, n-1\}$ as corresponding elements of the universe with respect to the ordering of the universe, then $\rho(k, p, q, x, y)$ holds if and only if $h_{p,q}(x) = y$. Note that the p and q from the lemma could exceed n for very large X (they can reach up to $n^2 \log_2 n \leq n^3$), but, first, this situation will not arise in the following and, second, this could be fixed by using three variables to encode p and three variables to encode q. Trivially, $\rho(k, p, q, x, y)$ has some constant quantifier rank (the formula explicitly constructed by Chen et al. has $\mathrm{qr}(\rho) = 9$, assuming $k^2 < n$).

Next, we will need the basic idea or "trick" of Alon et al.'s [3] color coding technique: While for appropriate p and q the function $h_{p,q}$ will "just" be injective on $\{0,\dots,k^2-1\}$, we actually want a function that maps each element $x \in X$ to a specific element ("the color of x") of $\{1,\dots,k\}$. Fortunately, this is easy to achieve by concatenating $h_{p,q}$ with an appropriate function $g\colon \{0,\dots,k^2-1\} \to \{1,\dots,k\}$.

In detail, to construct ϕ' from the claim of the theorem, we construct a family of formulas $\phi^g(p,q)$ where p and q are new free variables and the formulas are indexed by all possible functions $g\colon \{0,\dots,k^2-1\} \to \{1,\dots,k\}$: In ϕ, replace every occurrence of $C_i x_j$ by the following formula $\pi_i^g(p,q,x_j)$:

$$\bigvee_{y\in\{0,\dots,k^2-1\},g(y)=i} \exists\hat{k}\exists\hat{y}\big(\textsc{succ}^k(0)=\hat{k}\wedge\textsc{succ}^y(0)=\hat{y}\wedge\rho(\hat{k},p,q,x_j,\hat{y})\big)$$

where $\hat{k}$ and $\hat{y}$ are fresh variables that we bind to the numbers k and y (if the universe is large enough). Note that the formula $C_i x_j$ has x_j as a free variable, while $\pi_i^g(p,q,x_j)$ additionally has p and q as free variables. As an example, for the formula $\phi = \exists x(C_2 x \vee \exists y C_5 y)$ we would have $\phi^g = \exists x(\pi_2^g(p,q,x) \vee \exists y \pi_5^g(p,q,y))$. Clearly, each ϕ^g has the property $\mathrm{qr}(\phi^g) = \mathrm{qr}(\phi) + O(1)$.

The desired formula ϕ' is (almost) simply $\bigvee_{g:\{0,\dots,k^2-1\}\to\{1,\dots,k\}} \exists p\exists q(\phi^g(p,q))$. The "almost" is due to the fact that this formula works only for structures with a sufficiently large universe—but by Lemma 1 it suffices to consider only this case. Let us prove that for every σ-structure $\mathcal{A}$ with universe $A = \{0,\dots,n-1\}$ and $n \geq c$ for some to-be-specified constant c, the following two statements are equivalent:

1. There is a k-coloring $\mathcal{B}$ of $\mathcal{A}$ with $\mathcal{B} \models \phi$.
2. $\mathcal{A} \models \bigvee_{g:\{0,\dots,k^2-1\}\to\{1,\dots,k\}} \exists p\exists q(\phi^g(p,q))$.

Let us start with the implication of item 2 to 1. Suppose there is a function $g\colon \{0,\dots,k^2-1\} \to \{1,\dots,k\}$ and elements $p,q \in \{0,\dots,n-1\}$ such that $\mathcal{A} \models \phi^g(p,q)$. We define a partition $A_1 \dot\cup \cdots \dot\cup A_k = A$ by $A_i = \{x \in A \mid g(h_{p,q}(x)) = i\}$. In other words, A_i contains all elements of A that are first hashed to an element of $\{0,\dots,k^2-1\}$ that is then mapped to i by the function g. Trivially, the A_i form a partition of the universe A.

Assuming that the universe size is sufficiently large, namely for $k^2 \log_2 n < n$, inside ϕ^g all uses of $\rho(\hat{k},p,q,x,\hat{y})$ will have the property that $\mathcal{A} \models \rho(\hat{k},p,q,x,\hat{y})$ if and only if $h_{p,q}(x) = \hat{y}$. Clearly, there is a constant c depending only on k such that for all $n > c$ we have $k^2 \log_2 n < n$.

With the property established, we now see that $\pi_i^g(p,g,x_j)$ holds inside the formula ϕ^g if and only if the interpretation of x_j is an element of A_i. This means that if we interpret each C_i by A_i, then we get $\mathcal{A} \models \phi(A_1,\dots,A_k)$ and the A_i form a partition of the universe. In other words, we get item 1.

Now assume that item 1 holds, that is, there is a partition $B_1 \dot\cup \cdots \dot\cup B_k = A$ with $\mathcal{A} \models \phi(B_1,\dots,B_k)$. We must show that there are a $g\colon \{0,\dots,k^2-1\} \to \{1,\dots,k\}$ and $p,q \in A$ such that $\mathcal{A} \models \phi^g(p,q)$.

At this point, we use the small witness property that we established earlier for the partition. By this property there are pairwise disjoint sets $A_i \subseteq A$ such that, first, $|A_i|$ depends only on ϕ and, second, $\mathcal{A} \models \phi(A_1,\dots,A_k)$. Let $X = \bigcup_{i=1}^k A_i$. Then $|X|$ depends only on ϕ and let s_ϕ be a ϕ-dependent upper bound on this size. By the universal hashing lemma (Lemma 3), there are now p and q such that $h_{p,q}\colon \{0,\dots,n-1\} \to \{0,\dots,s_\phi^2-1\}$ is injective on X. Then, we can set $g\colon \{0,\dots,s_\phi^2-1\} \to \{1,\dots,k\}$ to $g(v) = i$ if there is an $x \in A_i$ with $h_{p,q}(x) = v$ and setting $g(v)$ arbitrarily otherwise. Note that this is, indeed, a valid definition of g since $h_{p,q}$ is injective on X.

With these definitions, we now define the following sets D_1 to D_k: Let $D_i = \{x \in A \mid g(h_{p,q}(\hat{x})) = i\}$ where $\hat{x}$ is the index of x in A with respect to the ordering (that is, $\hat{x} = |\{y \in A \mid y <^{\mathcal{A}} x\}|$ and for the special case that $A = \{0,\dots,n-1\}$ and that $<^{\mathcal{A}}$ is the natural ordering, $\hat{x} = x$). Observe that $D_i \supseteq A_i$ holds for all D_i and that the D_i form a partition of the universe A. By the monotonicity property, $\mathcal{A} \models \phi(A_1,\dots,A_k)$

implies $\mathcal{A} \models \phi(D_1, \ldots, D_k)$. However, by definition of the D_i and of the formulas π_i^g, for a sufficiently large universe size n (namely $s_\phi^2 \log_2 n < n$), we now also have $\mathcal{A} \models \phi^g(p, q)$, which in turn implies $\mathcal{A} \models \bigvee_g \exists p \exists q \phi^g$.

This concludes the proof of Theorem 2. □

In the theorem we assumed that ϕ is a sentence (a formula without free variables) to keep the notation simple, but both the theorem and later theorems still hold when $\phi(x_1, \ldots, x_n)$ has free variables x_1 to x_n. Then there is a corresponding $\phi'(x_1, \ldots, x_n)$ such that first item becomes that for all $\mathcal{A} \in \text{STRUC}[\tau]$ and all $a_1, \ldots, a_n \in |\mathcal{A}|$ we have $\mathcal{A} \models \phi'(a_1, \ldots, a_n)$ if and only if there is a k-coloring $\mathcal{B}$ of $\mathcal{A}$ with $\mathcal{B} \models \phi(a_1, \ldots, a_n)$. Note that the syntactic transformations in the theorem do not add dependencies of universal quantifiers on the free variables.

Mostly for simplicity, we have opted to only use first-order logic throughout this paper and this will be exactly the kind of logic we need in the rest of this paper. However, it is arguably more natural to formulate the above Theorem 2 in terms of second-order logic since "guessing a coloring" is clearly a case of a special existential second-order quantification. The below corollary is a rephrasing of Theorem 2 in terms of second-order logic. In the corollary, we use the notation $\exists(C_1 \dot\cup \cdots \dot\cup C_k)(\phi)$ as a shorthand for $\exists C_1 \cdots \exists C_k(\phi \wedge \forall x \bigwedge_{i \neq j}(\neg C_i x \wedge \neg C_j x))$, that is, for the "existential guessing of a coloring where each element of the universe gets at most one color".

Corollary 1. *Let τ be an arithmetic signature and let k be a number. For each first-order $\tau_{k\text{-colors}}$-sentence ϕ in negation normal form in which no C_i is inside a universal scope, there is a first-order τ-sentence ϕ' such that:*

1. $\exists(C_1 \dot\cup \cdots \dot\cup C_k)(\phi) \equiv \phi'$ *(on finite structures).*
2. $\text{qr}(\phi') = \text{qr}(\phi) + O(1)$.
3. $|\text{bound}(\phi')| = |\text{bound}(\phi)| + O(1)$.

3.2. Formulas with Weak Quantifiers

If one has a closer look at proofs based on color coding, one cannot help but notice that the colors are almost exclusively used to ensure that certain vertices in a structure are distinct from certain other vertices: Recall the introductory example $\bigwedge_{j=1}^k \exists x \exists y \exists z(Exy \wedge Eyz \wedge Exz \wedge C_{3j-2}x \wedge C_{3j-1}y \wedge C_{3j}z)$, which describes the triangle packing problem when we require that the C_i form a partition of the universe. Since the C_i are only used to ensure that the many different x, y, and z are different, we already rewrote the formula in (4) as $\exists x_1 \cdots \exists x_{3k} \bigwedge_{i \neq j} x_i \neq x_j \wedge \bigwedge_{j=1}^k \exists x \exists y \exists z(Exy \wedge Eyz \wedge Exz \wedge x_{3j-2} = x \wedge x_{3j-1} = y \wedge x_{3j} = z)$. While this rewriting gets rid of the colors and moves us back into the familiar territory of simple first-order formulas, the quantifier rank and the number of variables in the formula have now "exploded" (from the constant 3 to the parameter-dependent value $3k + 3$)—which is exactly what we need to avoid in order to apply Fact 1 or Theorem 1.

We now define a syntactic property that the x_i have that allows us to remove them from the formula and, thereby, to arrive at a family of formulas of constant quantifier rank. For a (sub)formula α of the form $\forall d(\phi)$ or $\exists d(\phi)$, we say that d depends on all free variables in ϕ (at the position of α in a larger formula). For instance, in $Exy \wedge \forall b(Exb \wedge \exists z(Eyz)) \wedge \exists b(Exx)$, the variable b depends on x and y at its first binding ($\forall b$) and on x at the second binding ($\exists b$).

Definition 3. *Let ϕ be in negation normal form. We call the leading existential quantifier of $\exists x(\phi)$ strong if*

1. *some universal binding inside ϕ depends on x or*
2. *there is a subformula $\alpha \wedge \beta$ of ϕ such that both α and β contain x in literals that are not of the form $x \neq y$ for some variable y.*

Dually, we call the universal leading quantifier of $\forall x(\phi)$ strong if

1. *some existential binding inside ϕ depends on x or*

2. *there is a subformula $\alpha \vee \beta$ such that both α and β contain x in literals that are not of the form $x = y$ for some variable y.*

A quantifier leading a (sub)formula that is not strong is weak. The strong quantifier rank strong-qr(ψ) *is the quantifier rank of ψ, where weak quantifiers are ignored;* strong-bound(ψ) *contains all variables of ψ that are bound by non-weak quantifiers.*

We place a dot on the variables bound by weak quantifiers to make them easier to spot. For example, in $\phi = \exists x \exists y \exists \dot{z}(Rxx\dot{z}\dot{z} \wedge x \neq y \wedge y \neq \dot{z} \wedge Px \wedge \forall \dot{w}\, E\dot{w}yy)$ the quantifier $\exists \dot{z}$ is weak, but neither are $\exists x$ (since x is used in two literals joined by a conjunction, namely in $Rxx\dot{z}\dot{z}$ and Px) nor $\exists y$ (since $\dot{w}$ depends on y in $\forall \dot{w}\, E\dot{w}yy$). We have $\mathrm{qr}(\phi) = 4$, but strong-qr$(\phi) = 2$, and bound$(\phi) = \{x, y, \dot{z}, \dot{w}\}$, but strong-bound$(\phi) = \{x, y\}$.

Admittedly, the definition of weakness is a bit technical, but note that there is a rather simple sufficient condition for an existentially-bound variable x to be weak: If it is not used in universal bindings and is used in only one literal that is not an inequality, then x is weak. This condition almost always suffices for identifying the weak variables, although there are of course exceptions like $\exists \dot{x}(P\dot{x} \vee Q\dot{x})$.

Our objective is now to simultaneously remove all weak quantifiers from a formula without increasing the strong quantifier rank by more than a constant factor or the number of strong variables by more than a constant. We first prove this only for existential weak quantifiers in the below theorem (by considering only formulas that do not have weak universal quantifiers). Once we have achieved this, a comparatively simple syntactic duality argument allows us to extend the claim to all formulas in Theorem 4.

Theorem 3. *Let τ be an arithmetic signature. Then for every τ-formula ϕ in negation normal form without weak universal quantifiers there is a τ-formula ϕ' such that*

1. *$\phi' \equiv \phi$ (on finite structures),*
2. *$\mathrm{qr}(\phi') = 3 \cdot$ strong-qr$(\phi) + \mathrm{arity}(\tau) + O(1)$, and*
3. *$|\mathrm{bound}(\phi')| = |$strong-bound$(\phi)| + \mathrm{arity}(\tau) + O(1)$.*

Before giving the detailed proof, we briefly sketch the overall idea: Using simple syntactic transformations, we can ensure that all weak quantifiers follow in blocks after universal quantifiers (and, by assumption, all universal quantifiers are strong). We can also ensure that inequality literals directly follow the blocks of weak quantifiers and are joined by conjunctions. If the inequality literals following a block happen to require that all weak variables from the block are different (that is, if for all pairs $\dot{x}_i$ and $\dot{x}_j$ of different weak variables there is an inequality $\dot{x}_i \neq \dot{x}_j$), then we can remove the weak quantifiers $\exists \dot{x}_i$ and at the (typical single) place where $\dot{x}_i$ is used, we use a color C_i instead. For instance, if $\dot{x}_i$ is used in the literal $\dot{x}_i = y$, we replace the literal by $C_i y$. If $\dot{x}_i$ is used for instance in $\neg E\dot{x}_i y$, we replace this by $\exists x(C_i x \wedge \neg Exy)$. In this way, for each block we get an equivalent formula to which we can apply Theorem 2. A more complicated situation arises when the inequality literals in a block "do not require complete distinctness," but this case can also be handled by considering all possible ways in which the inequalities can be satisfied in parallel. In result, all weak quantifiers get removed and for each block a constant number of new quantifiers are introduced. Since each block follows a different universal quantifier, the new total quantifier rank is at most the strong quantifier rank times a constant factor; and the new number of variables is only a constant above the number of original strong variables.

Proof of Theorem 3. Let ϕ be given. We first apply a number of simple syntactic transformations to move the weak quantifiers directly behind universal quantifiers and to move inequality literals directly behind blocks of weak quantifiers. Then we show how sets of inequalities can be "completed" if necessary. Finally, we inductively transform the formula in such a way that Theorem 2 can be applied repeatedly. As a running example

of how the different syntactic transformations work, we use the (semantically not very sensible) formula

$$\phi = \exists a\big(\exists \dot{x}(Ea\dot{x} \wedge \exists \dot{y}(\dot{y} \neq \dot{x})) \wedge \forall c \exists \dot{x} \exists \dot{y}(E\dot{x}\dot{y} \vee \exists z(\dot{x} \neq \dot{y} \wedge Pz \wedge Qcaz))\big). \tag{6}$$

In this formula, the (only) universally quantified variable, c, is strong since the existential binding $\exists z$ depends on it via $Qcaz$. The variable a is strong since $\forall c$ depends in it, once more via $Qcaz$. Finally, z is strong since it is used is two parts of a conjunction: $Pz \wedge Qcaz$.

Preliminaries: It will be useful to require that all weak variables are different. Thus, as long as necessary, when a variable is bound by a weak quantifier and once more by another quantifier, replace the variable used by the weak quantifier by a fresh variable. Note that this may increase the number of distinct (weak) variables in the formula, but we will get rid of all of them later on anyway. From now on, we may assume that the weak variables are all distinct from one another and also from all other variables.

It will also be useful to assume that ϕ starts with a universal quantifier. If this is not the case, replace ϕ by the equivalent formula $\forall v(\phi)$ where v is a fresh variable. This increases the quantifier rank by at most 1.

Finally, it will also be useful to assume that the formula has been "flattened" as in the proof Theorem 1 (one can loosely think of this as bringing the formula "locally into disjunctive normal form"): We use the distributive laws of propositional logic to repeatedly replace subformulas of the form $(\alpha \vee \beta) \wedge \gamma$ by $(\alpha \wedge \gamma) \vee (\beta \wedge \gamma)$ and $\alpha \wedge (\beta \vee \gamma)$ by $(\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$. Note that this transformation does not change which variables are weak.

For our running example, applying the described preprocessing yields:

$$\phi \equiv \forall v \exists a\big(\exists \dot{x}_1(Ea\dot{x}_1 \wedge \exists \dot{x}_2(\dot{x}_2 \neq \dot{x}_1)) \wedge \forall c \exists \dot{x}_3 \exists \dot{x}_4(E\dot{x}_3\dot{x}_4 \vee \exists z(\dot{x}_3 \neq \dot{x}_4 \wedge Pz \wedge Qcaz))\big).$$

Syntactic transformations I: blocks of weak quantifiers. The first interesting transformation is the following: We wish to move weak quantifiers "as far up the syntax tree as possible." To achieve this, we apply the following equivalences as long as possible by always replacing the left-hand side (and also commutatively equivalent formulas) by the right-hand side:

$$\begin{aligned} \exists \dot{x}(\alpha) \wedge \beta &\equiv \exists \dot{x}(\alpha \wedge \beta), \\ \exists \dot{x}(\alpha) \vee \beta &\equiv \exists \dot{x}(\alpha \vee \beta), \\ \exists y \exists \dot{x}(\alpha) &\equiv \exists \dot{x} \exists y(\alpha). \end{aligned}$$

Note that β does not contain $\dot{x}$ since we made all weak variables distinct and, of course, by $\exists y$ we mean a strong quantifier.

Once the transformations have been applied exhaustively, all weak quantifiers will be directly preceded in ϕ by either a universal quantifier or another weak quantifier. This means that all weak quantifiers are now arranged in blocks inside ϕ, each block being preceded by a universal quantifier.

$$\phi \equiv \forall v \exists \dot{x}_1 \exists \dot{x}_2 \exists a\big(Ea\dot{x}_1 \wedge \dot{x}_2 \neq \dot{x}_1 \wedge \forall c \exists \dot{x}_3 \exists \dot{x}_4(E\dot{x}_3\dot{x}_4 \vee \exists z(\dot{x}_3 \neq \dot{x}_4 \wedge Pz \wedge Qcaz))\big)$$

Syntactic transformations II: weak and strong literals. In order to apply color coding later on, it will be useful to have only three kinds of literals in ϕ:

1. Strong literals are literals that do not contain any weak variables.
2. Weak equalities are literals of the form $\dot{x} = y$ involving exactly one strong variable that is existentially bound inside the weak variable's scope:

$$\exists \dot{x}(\dots \exists y(\dots \dot{x} = y \dots) \dots).$$

3. Weak inequalities are literals of the form $\dot{x} \neq \dot{y}$ for two weak variables.

Let us call all other kinds of literals bad. This includes literals like $E\dot{x}\dot{x}$ or $Ez\dot{y}$ that contain a relation symbol and some weak variables, but also inequalities $\dot{x} \neq y$ involving a weak and a strong variable, equalities $\dot{x} = \dot{y}$ involving two weak variables, or an equality literal like the one in $\forall y \exists \dot{x}(\dot{x} = y)$. Finally, literals involving the successor function and weak variables are also bad.

In order to get rid of the bad literals, we will replace them by equivalent formulas that do not contain any bad literals. The idea is that we bind the variable or term that we wish to get rid of using a new existential quantifier. In order to avoid introducing too many new variables, for all of the following transformations we use the set of fresh variables v_1, v_2, and so on, where we may need more than one of these variables per literal, but will need no more than $O(\text{arity}(\tau))$ (recall that $\text{arity}(\tau)$ is the maximum arity of relation symbols in τ).

Let λ be a bad literal, that is, let it contain a weak variable $\dot{x}$, but neither be a weak equality nor a weak inequality. Replace λ by $\exists v_i(v_i = \dot{x} \wedge \lambda[\dot{x} \hookrightarrow v_i])$. Here, $\lambda[t_1 \hookrightarrow t_2]$ is our notation for the substitution of the term t_1 by t_2 in λ. The number i is chosen minimally so that λ does not already contain v_i. Since this transformation reduces the number of weak variables in λ and does not introduce a bad literal ($v_i = \dot{x}$ is a weak equality and hence "good"), sooner or later we will have gotten rid of all bad literals. For each literal we use at most $\text{arity}(\tau)$ new variables from the v_i (more precisely, at most $\max\{\text{arity}(\tau), 2\}$ in case τ contains only unary or no relation symbols and λ is something like $\text{SUCC}(\dot{x}) = \text{SUCC}(\dot{y})$, causing two replacements). Overall, we get that ϕ is equivalent to a formula without any bad literals in which we use at most $\text{arity}(\tau) + 2 = \text{arity}(\tau) + O(1)$ additional variables and whose quantifier rank is larger than that of ϕ by at most $\text{arity}(\tau) + O(1)$. Note that the transformation ensures that weak variables stay weak. Applied to our example formula, we get:

$$\forall v \exists \dot{x}_1 \exists \dot{x}_2 \exists a \big(\exists v_1(v_1 = \dot{x}_1 \wedge Eav_1) \wedge \dot{x}_2 \neq \dot{x}_1 \wedge \\ \forall c \exists \dot{x}_3 \exists \dot{x}_4 (\exists v_1(v_1 = \dot{x}_3 \wedge \exists v_2(v_2 = \dot{x}_4 \wedge Ev_1v_2)) \vee \exists z(\dot{x}_3 \neq \dot{x}_4 \wedge Pz \wedge Qcaz))\big)$$

Syntactic transformations III: accumulating weak inequalities. We now wish to move all weak inequalities to the "vicinity" of the corresponding block of weak quantifiers. More precisely, just as we did earlier, we apply the following equivalences (interpreted once more as rules that are applied from left to right):

$$(\dot{x} \neq \dot{y} \vee \alpha) \wedge \beta \equiv (\dot{x} \neq \dot{y} \wedge \beta) \vee (\alpha \wedge \beta), \tag{7}$$

$$\exists x(\alpha \vee \beta) \equiv \exists x(\alpha) \vee \exists x(\beta), \tag{8}$$

$$\exists z(\dot{x} \neq \dot{y} \wedge \alpha) \equiv \dot{x} \neq \dot{y} \wedge \exists z(\alpha). \tag{9}$$

Note that these rules do not change which variables are weak. When these rules can no longer be applied, the weak inequality are "next" to their quantifier block, that is, each subformula starting with weak quantifiers has the form

$$\exists \dot{x}_{i_1} \cdots \exists \dot{x}_{i_k} \bigvee_i \big(\big(\bigwedge_j \lambda_i^j \big) \wedge \alpha_i \big)$$

where the α_i contain no weak inequalities while all λ_i^j are weak inequalities.

For our example formula, we get:

$$\phi \equiv \forall v \exists \dot{x}_1 \exists \dot{x}_2 \big(\dot{x}_2 \neq \dot{x}_1 \wedge \exists a \exists v_1(v_1 = \dot{x}_1 \wedge Eav_1) \wedge \\ \forall c \big(\exists \dot{x}_3 \exists \dot{x}_4 (\exists v_1(v_1 = \dot{x}_3 \wedge \exists v_2(v_2 = \dot{x}_4 \wedge Ev_1v_2))) \vee \\ (\dot{x}_3 \neq \dot{x}_4 \wedge \exists z(Pz \wedge Qcaz))\big)\big).$$

Finally, we now swap each block of weak quantifiers with the following disjunction, that is, we apply the following equivalence from left to right:

$$\exists \dot{x}_{i_1} \cdots \exists \dot{x}_{i_k} \bigvee_i \psi_i \equiv \bigvee_i \exists \dot{x}_{i_1} \cdots \exists \dot{x}_{i_k} \psi_i.$$

If necessary, we rename weak variables to ensure once more that they are unique. For our example, the different transformations yield:

$$\begin{aligned}\phi \equiv \forall v \exists \dot{x}_1 \exists \dot{x}_2 \big(\dot{x}_2 \neq \dot{x}_1 \wedge \exists a \exists v_1 (v_1 = \dot{x}_1 \wedge Eav_1) \wedge \\ \forall c \big(\exists \dot{x}_3 \exists \dot{x}_4 (\exists v_1 (v_1 = \dot{x}_3 \wedge \exists v_2 (v_2 = \dot{x}_4 \wedge Ev_1 v_2))) \vee \\ \exists \dot{x}_5 \exists \dot{x}_6 (\dot{x}_5 \neq \dot{x}_6 \wedge \exists z (Pz \wedge Qcaz))) \big).\end{aligned}$$

Let us spell out the different ψ_i, λ_i^j, and α_i contained in the above formula: First, there is one block of weak variables ($\exists \dot{x}_1 \exists \dot{x}_2$) following $\forall v$ at the beginning. There is only a single formula ψ_1 for this block, which equals $(\bigwedge_{j=1}^1 \lambda_1^j) \wedge \alpha_1$ with $\lambda_1^1 = (\dot{x}_2 \neq \dot{x}_1)$ and $\alpha_1 = \exists a \exists v_1 (v_1 = \dot{x}_1 \wedge Eav_1) \wedge \forall c(\dots)$. Second, there are two blocks of weak variables ($\exists \dot{x}_3 \exists \dot{x}_4$ and $\exists \dot{x}_5 \exists \dot{x}_6$) following $\forall c$, which are followed by (new) formulas ψ_1 and ψ_2. The first is of the form $\psi_1 = (\bigwedge_{j=1}^0 \lambda_1^j) \wedge \alpha_1$ and the second of the form $\psi_2 = (\bigwedge_{j=1}^1 \lambda_2^j) \wedge \alpha_2$. There are no λ_1^j and we have $\alpha_1 = \exists v_1 (v_1 = \dot{x}_3 \wedge \exists v_2 (v_2 = \dot{x}_4 \wedge Ev_1 v_2))$. We have $\lambda_2^1 = (\dot{x}_5 \neq \dot{x}_6)$ and we have $\alpha_2 = \exists z (Pz \wedge Qcaz)$.

We make the following observation at this point: Inside each ψ_i, each of the variables $\dot{x}_{i_1}$ to $\dot{x}_{i_k}$ is used at most once outside of weak inequalities. The reason for this is that rules (7) and (8) ensure that there are no disjunctions inside the ψ_i that involve a weak variable $\dot{x}$. Thus, the requirement "in any subformula of ψ_i of the form $\alpha \wedge \beta$ only α or β—but not both—may use $\dot{x}$ in a literal that is not a weak inequality" from the definition of weak variables just boils down to "$\dot{x}$ may only be used once in ψ_i in a literal that is not a weak inequality." Since weak variables cannot be present in strong literals, this means, in particular, that $\dot{x}$ is now only used in (at most) a single weak equality $\dot{x} = y$ and otherwise only in weak inequalities.

Syntactic transformations IV: completing weak inequalities. The last step before we can apply the color coding method is to "complete" the conjunctions of weak inequalities. After all the previous transformations have been applied, each block of weak quantifiers has now the form $\exists \dot{x}_1 \cdots \exists \dot{x}_k \big(\bigwedge_i \lambda_i \wedge \alpha \big)$ where the λ_i are all weak inequalities (between some or all pairs of $\dot{x}_1$ to $\dot{x}_k$) and α contains no weak inequalities involving the $\dot{x}_i$ (but may contain one weak equality for each $\dot{x}_i$). Of course, the weak variables need not be $\dot{x}_1$ to $\dot{x}_k$, but let us assume this to keep to notation simple.

The formula $\bigwedge_i \lambda_i$ expresses that some of the variables $\dot{x}_i$ must be different. If the formula encompasses all possible weak inequalities between distinct $\dot{x}_i$ and $\dot{x}_j$, then the formula would require that all $\dot{x}_i$ must be distinct—exactly the situation in which color coding can be applied. However, some weak inequalities may be "missing" such as in the formula $\dot{x}_1 \neq \dot{x}_2 \wedge \dot{x}_2 \neq \dot{x}_3 \wedge \dot{x}_1 \neq \dot{x}_3 \wedge \dot{x}_3 \neq \dot{x}_4$: This formula requires that $\dot{x}_1$ to $\dot{x}_3$ must be distinct and that $\dot{x}_4$ must be different from $\dot{x}_3$—but it would be allowed that $\dot{x}_4$ equals $\dot{x}_1$ or $\dot{x}_2$. Indeed, it might be the case that the only way to make α true is to make $\dot{x}_1$ equal to $\dot{x}_4$. This leads to a problem in the context of color coding: We want to color $\dot{x}_1$, $\dot{x}_2$, and $\dot{x}_3$ differently, using, say, red, green, and blue. In order to ensure $\dot{x}_3 \neq \dot{x}_4$, we must give $\dot{x}_4$ a color different from blue. However, it would be wrong to color it red or green or using a new color like yellow since each would rule out $\dot{x}_4$ being equal or different from either $\dot{x}_1$ or $\dot{x}_2$—and each possibility must be considered to ensure that we miss no assignment that makes α true.

The trick at this point is to reduce the problem of missing weak inequalities to the situation where all weak inequalities are present by using a large disjunction over all possible ways to unify weak variables without violating the weak inequalities.

In detail, let us call a partition $P_1 \dot{\cup} \cdots \dot{\cup} P_l$ of the set $\{\dot{x}_1, \dots, \dot{x}_k\}$ allowed by the λ_i if the following holds: For each P_j and any two different $\dot{x}_p, \dot{x}_q \in P_j$ none of the λ_i is the

inequality $\dot{x}_p \neq \dot{x}_q$. In other words, the λ_i do not forbid that the elements of any P_j are identical. Clearly, the partition with $P_j = \{\dot{x}_j\}$ is always allowed by any λ_i, but in the earlier example, the partition $P_1 = \{\dot{x}_1, \dot{x}_4\}, P_2 = \{\dot{x}_2\}, P_3 = \{\dot{x}_3\}$ would be allowed, while $P_1 = \{\dot{x}_1\}, P_2 = \{\dot{x}_2\}, P_3 = \{\dot{x}_3, \dot{x}_4\}$ would not be.

We introduce the following notation: For a partition $P_1 \dot{\cup} \cdots \dot{\cup} P_l = \{\dot{x}_1, \ldots, \dot{x}_k\}$ we will write distinct$(P_1, \ldots, P_l)$ for $\bigwedge_{1 \leq i < j \leq l, \dot{x}_p \in P_i, \dot{x}_q \in P_j} \dot{x}_p \neq \dot{x}_q$. We claim the following:

Claim 1. *For any weak inequalities λ_i we have*

$$\bigwedge_i \lambda_i \equiv \bigvee_{P_1 \dot{\cup} \cdots \dot{\cup} P_l \text{ is allowed by the } \lambda_i} \text{distinct}(P_1, \ldots, P_l).$$

Proof. For the implication from left to right, assume that $\mathcal{A} \models \bigwedge_i \lambda_i(a_1, \ldots, a_k)$ for some (not necessarily distinct) $a_1, \ldots, a_k \in |\mathcal{A}|$. The elements induce a natural partition $P_1 \dot{\cup} \cdots \dot{\cup} P_l = \{\dot{x}_1, \ldots, \dot{x}_k\}$ where two variables $\dot{x}_p$ and $\dot{x}_q$ are in the same set P_j if and only if $a_p = a_q$. Then, clearly, for all i and j with $1 \leq i < j \leq l$ and any $\dot{x}_p \in P_i$ and $\dot{x}_q \in P_j$ we have $a_i \neq a_j$. Thus, all inequalities in distinct$(P_1, \ldots, P_l)$ are satisfied and, hence, the right-hand side holds.

For the other direction, suppose that $\mathcal{A}$ is a model of the right hand side for some a_1 to a_k. Then there must be a partition $P_1 \dot{\cup} \cdots \dot{\cup} P_l$ that is allowed by the λ_i such that $\mathcal{A}$ is also a model of distinct$(P_1, \ldots, P_l)$. Furthermore, each λ_i is actually present in this last formula: If $\dot{x}_p \neq \dot{x}_q$ is one of the λ_i, then by the very definition of "$P_1 \dot{\cup} \cdots \dot{\cup} P_l$ is allowed for the λ_i" we must have that $\dot{x}_p$ and $\dot{x}_q$ lie in different P_i and P_j—which, in turn, implies that $\dot{x}_p \neq \dot{x}_q$ is present in distinct$(P_1, \ldots, P_l)$. □

Applied to the example $\dot{x}_1 \neq \dot{x}_2 \wedge \dot{x}_2 \neq \dot{x}_3 \wedge \dot{x}_1 \neq \dot{x}_3 \wedge \dot{x}_3 \neq \dot{x}_4$ from above, the claim states the following: Since there are three partitions that are allowed by these literals (namely the one in which each variable gets its own equivalence class, the one where $\dot{x}_1$ and $\dot{x}_4$ are put into one class, and the one where $\dot{x}_2$ and $\dot{x}_4$ are put into one class) we have:

$$\begin{aligned} & \dot{x}_1 \neq \dot{x}_2 \wedge \dot{x}_2 \neq \dot{x}_3 \wedge \dot{x}_1 \neq \dot{x}_3 \wedge \dot{x}_3 \neq \dot{x}_4 \\ \equiv \quad & \text{distinct}(\{\dot{x}_1\}, \{\dot{x}_2\}, \{\dot{x}_3\}, \{\dot{x}_4\}) \\ & \vee \text{distinct}(\{\dot{x}_1, \dot{x}_4\}, \{\dot{x}_2\}, \{\dot{x}_3\}) \\ & \vee \text{distinct}(\{\dot{x}_1\}, \{\dot{x}_2, \dot{x}_4\}, \{\dot{x}_3\}). \end{aligned}$$

The claim has the following corollary:

Corollary 2. *For any weak inequalities λ_i involving only variables from $\{\dot{x}_1, \ldots, \dot{x}_k\}$ we have* $\exists \dot{x}_1 \cdots \exists \dot{x}_k (\bigwedge_i \lambda_i \wedge \alpha) \equiv \bigvee_{P_1 \dot{\cup} \cdots \dot{\cup} P_l \text{ is allowed by the } \lambda_i} \exists \dot{x}_1 \cdots \exists \dot{x}_k (\text{distinct}(P_1, \ldots, P_l) \wedge \alpha)$.

As in the previous transformations we now apply the equivalence from the corollary from left to right. If we create copies of α during this process, we rename the weak variables in these copies to ensure, once more, that each weak variable is unique. In our example formula ϕ, there is only one place where the transformation changes anything: The middle weak quantifier block (the $\exists \dot{x}_3 \exists \dot{x}_4$ block). For the first and the last block, the literals $\dot{x}_1 \neq \dot{x}_2$ and $\dot{x}_5 \neq \dot{x}_6$, respectively, already rule out all partitions except for the trivial one. For the middle block, however, there are no weak inequalities at all and, hence, there are now two allowed partitions: First, $P_1 = \{\dot{x}_3\}, P_2 = \{\dot{x}_4\}$, but also $P_1 = \{\dot{x}_3, \dot{x}_4\}$. This means that we get a copy of the middle block where $\dot{x}_3$ and $\dot{x}_4$ are required to be different—and we renumber them to $\dot{x}_7$ and $\dot{x}_8$:

$$\begin{aligned} \phi \equiv \forall v \exists \dot{x}_1 \exists \dot{x}_2 \big(& \dot{x}_2 \neq \dot{x}_1 \wedge \exists a \exists v_1 (v_1 = \dot{x}_1 \wedge Eav_1) \wedge \\ & \forall c \big(\exists \dot{x}_3 \exists \dot{x}_4 (\quad \exists v_1 (v_1 = \dot{x}_3 \wedge \exists v_2 (v_2 = \dot{x}_4 \wedge Ev_1v_2))) \vee \\ & \exists \dot{x}_7 \exists \dot{x}_8 (\dot{x}_7 \neq \dot{x}_8 \wedge \exists v_1 (v_1 = \dot{x}_7 \wedge \exists v_2 (v_2 = \dot{x}_8 \wedge Ev_1v_2))) \vee \\ & \exists \dot{x}_5 \exists \dot{x}_6 (\dot{x}_5 \neq \dot{x}_6 \wedge \exists z (Pz \wedge Qcaz))) \big). \end{aligned}$$

Of course, for our particular example, the introduction of $\dot{x}_7$ and $\dot{x}_8$ is superfluous insofar as it is not necessary to introduce the special case "force $\dot{x}_3$ and $\dot{x}_4$ to be different" in addition to the already present case "do not care whether $\dot{x}_3$ and $\dot{x}_4$ are different or not." It is only with more complex weak inequalities like $\dot{x}_1 \neq \dot{x}_2 \wedge \dot{x}_2 \neq \dot{x}_3 \wedge \dot{x}_1 \neq \dot{x}_3 \wedge \dot{x}_3 \neq \dot{x}_4$ that the syntactic transformation becomes really necessary.

Applying color coding: we are now ready to apply the color coding technique; more precisely, to repeatedly apply Theorem 2 to the formula ϕ. Before we do so, let us summarize the structure of ϕ:

1. All weak quantifiers come in blocks, and each such block either directly follows a universal quantifier or follows a disjunction after a universal quantifier. In particular, on any root-to-leaf path in the syntax tree of ϕ between any two blocks of weak quantifiers there is at least one universal quantifier.
2. All blocks of weak quantifiers have the form

$$\exists \dot{x}_{i_1} \cdots \exists \dot{x}_{i_k} \big(\text{distinct}(P_1, \ldots, P_l) \wedge \alpha\big) \tag{10}$$

for some partition $P_1 \dot{\cup} \cdots \dot{\cup} P_l = \{x_{i_1}, \ldots, x_{i_k}\}$ and for some α in which the only literals that contain any $\dot{x}_{i_j}$ are of the form $\dot{x}_{i_j} = y$ for a strong variable y that is bound by an existential quantifier inside α. Furthermore, none of these weak equality literals is in the scope of a universal quantifier inside α. (Of course, all variables in ϕ are in the scope of a universal quantifier since we added one at the start, but the point is that none of the $\dot{x}_i$ is in the scope of a universal quantifier that is inside α.)

In ϕ there may be several blocks of weak quantifiers, but at least one of them (let us call it β) must have the form (10) where α contains no weak variables other than $\dot{x}_{i_1}$ to $\dot{x}_{i_k}$. (For instance, in our example formula, this is the case for the blocks starting with $\exists \dot{x}_3 \exists \dot{x}_4$, for $\exists \dot{x}_7 \exists \dot{x}_8$, and for $\exists \dot{x}_5 \exists \dot{x}_6$, but not for $\exists \dot{x}_1 \exists \dot{x}_2$ since, here, the corresponding α contains all of the rest of the formula.) In our example, we could choose

$$\beta = \exists \dot{x}_7 \exists \dot{x}_8 (\dot{x}_7 \neq \dot{x}_8 \wedge \exists v_1 (v_1 = \dot{x}_7 \wedge \exists v_2 (v_2 = \dot{x}_8 \wedge E v_1 v_2)))$$

and would then have

$$\alpha = \exists v_1 (v_1 = \dot{x}_7 \wedge \exists v_2 (v_2 = \dot{x}_8 \wedge E v_1 v_2)).$$

We build a new formula α' from α as follows: We replace each occurrence of a weak equality $\dot{x}_i = y$ in α for some weak variable $\dot{x}_i \in P_j$ and some strong variable y by the formula $C_j y$. In our example, where $P_1 = \{\dot{x}_7\}$ and $P_2 = \{\dot{x}_8\}$ we would get

$$\alpha' = \exists v_1 (C_1 v_1 \wedge \exists v_2 (C_2 v_2 \wedge E v_1 v_2)).$$

An important observation at this point is that α' contains no weak variables any longer, while no additional variables have been added. In particular, the quantifier rank of α' equals the strong quantifier rank of α and the number of variables in α' equals the number of strong variables in α.

Note that the literals $C_j y$ and also $\dot{x}_i = y$ are positive since the formulas are in negation normal form. Hence, they have the following monotonicity property: If some structure together with some assignment to the free variables is a model of α or α', but a literal $\dot{x}_i = y$ or $C_j y$ is false, the structure will still be a model if we replace the literal by a tautology.

For simplicity, in the following, we assume that $\dot{x}_{i_1}$ to $\dot{x}_{i_k}$ are just $\dot{x}_1$ to $\dot{x}_k$. Additionally, for simplicity we assume that β contains no free variables when, in fact, it can. However, these variables cannot be any of the variables y for which we make changes and, thus, it keeps the notation simpler to ignore the additional free variables here. The following statement simply holds for all assignments to them:

Claim 2. *Let $P_1 \dot{\cup} \cdots \dot{\cup} P_l = \{x_1, \ldots, x_k\}$. Then for each structure $\mathcal{A}$, the following are equivalent:*

1. $\mathcal{A} \models \exists \dot{x}_1 \cdots \exists \dot{x}_k \big(\text{distinct}(P_1, \ldots, P_l) \wedge \alpha\big)$.
2. *There are elements* $a_1, \ldots, a_k \in |\mathcal{A}|$ *with* $\mathcal{A} \models \alpha(a_1, \ldots, a_k)$ *and such that* $a_p \neq a_q$ *whenever* $\dot{x}_p \in P_i$, $\dot{x}_q \in P_j$, *and* $i \neq j$.
3. *There is an* l*-coloring* $\mathcal{B}$ *of* $\mathcal{A}$ *such that* $\mathcal{B} \models \alpha'$.

Proof. For the proof of the claim, it will be useful to apply some syntactic transformations to α and α'. Just like the many transformations we encountered earlier, these transformations yield equivalent formulas and, thus, it suffices to prove the claim for them (since the claim is about the models of α and α'). However, these transformations are needed only to prove the claim, they are not part of the "chain of transformations" that is applied to the original formula (they increase the number of strong variables far too much).

In α there will be some occurrences of literals of the form $\dot{x}_i = y$. For each such occurrence, there will be exactly one subformula in α of the form $\exists y(\gamma)$ where γ contains $\dot{x}_i = y$. We now apply two syntactic transformations: First, we replace y in $\exists y(\gamma)$ by a fresh new variable y_i (that is, we replace all free occurrences of y inside γ by y_i and we replace the leading $\exists y$ by $\exists y_i$). Second, we "move all $\exists y_i$ to the front" by simply deleting all occurrences of $\exists y_i$ from α, resulting in a formula δ, and then adding the block $\exists y_1 \cdots \exists y_k$ before δ. As an example, if we apply these transformations to $\alpha = \exists v_1(\dot{x}_7 = v_1 \wedge \exists v_2(\dot{x}_8 = v_2 \wedge Ev_1v_2))$, the first transformation yields

$$\exists y_7(\dot{x}_7 = y_7 \wedge \exists y_8(\dot{x}_8 = y_8 \wedge Ey_7y_8))$$

and the second one yield the new

$$\alpha = \exists y_1 \cdots \exists y_8(\dot{x}_7 = y_7 \wedge \dot{x}_8 = y_8 \wedge Ey_7y_8).$$

In α', we apply exactly the same transformations, only now the literals we look for are not $\dot{x}_i = y$, but $C_j y$. We still apply the same renaming of y (namely to y_i and not to y_j) as in α and apply the same movement of the quantifiers. This results in a new formula α' of the form $\exists y_1 \cdots \exists y_k(\delta')$. For $\alpha' = \exists v_1(C_1v_1 \wedge \exists v_2(C_2v_2 \wedge Ev_1v_2))$ we get the new

$$\alpha' = \exists y_1 \cdots \exists y_8(C_1y_7 \wedge C_2y_8 \wedge Ey_7y_8)$$

and δ' is now the inner part without the quantifiers.

Let us now prove the claim. The first two items are trivially equivalent by the definition of $\text{distinct}(P_1, \ldots, P_l)$.

The second statement implies the third: To show this, for $j \in \{1, \ldots, l\}$ we first set $C_j^{\mathcal{B}} = \{a_i \mid \dot{x}_i \in P_j\}$ and then add $|\mathcal{A}| \setminus \{a_1, \ldots, a_k\}$ to, say, $C_1^{\mathcal{B}}$ in order to create a correct partition. This setting clearly ensures that whenever $\dot{x}_i = y$ holds in α, we also have $C_j y$ holding in α'. Since α' differs from α only on the literals of the form $\dot{x}_i = y$ (which got replaced by $C_j y$), since we just saw that when $\dot{x}_i = y$ holds in α, the replacements $C_j y$ holds in α', and since α has the monotonicity property (by which it does matter when more literals of the form $C_i y$ hold in α' than did in α), we get the third statement.

The third statement implies the second: Let an l-coloring $\mathcal{B}$ of $\mathcal{A}$ be given with $\mathcal{B} \models \alpha'$. Since $\alpha' = \exists y_1 \cdots \exists y_k(\delta')$, there must now be elements $b_1, \ldots, b_k \in |\mathcal{A}|$ such that $\mathcal{B} \models \delta'(b_1, \ldots, b_k)$. We define new elements $a_i \in |\mathcal{A}|$ as follows: If $b_i \in C_i^{\mathcal{B}}$, let $a_i = b_i$. Otherwise, let a_i be an arbitrary element of $C_i^{\mathcal{B}}$. We show in the following that the a_i constructed in this way can be used in the second statement, that is, we claim that $\mathcal{A} \models \alpha(a_1, \ldots, a_k)$ and the a_i have the distinctness property from the claim.

First, recall that α is of the form $\exists y_1 \cdots \exists y_k(\delta)$ (because of the syntactic transformations we applied for the purposes of the proof of this claim) and δ contains literals of the form $\dot{x}_i = y_i$, where the $\dot{x}_i$ are the free variables for which the values a_i are now plugged in. We claim that $\mathcal{A} \models \delta(a_1, \ldots, a_k, b_1, \ldots, b_k)$, that is, we claim that if we plug in a_1 to a_k for the free variables $\dot{x}_1$ to $\dot{x}_k$ in δ and we plug in b_1 to b_k for the (additional) free variables y_1 to y_k in δ, then δ holds in $\mathcal{A}$. To see this, recall that $\mathcal{B} \models \delta'(b_1, \ldots, b_k)$ holds and δ' is identical to δ except that $\dot{x}_i = y_i$ got replaced by $C_j y_i$. In particular, by construction of the a_i, whenever

$C_j y_i$ holds in $\mathcal{B}$ with y_i being set to b_i (that is, whenever $b_i \in C_j^{\mathcal{B}}$), we clearly also have that $\dot{x}_i = y_i$ holds in $\mathcal{A}$ with $\dot{x}_i$ being set to a_i and y_i being set to b_i (since we let $a_i = b_i$ whenever $b_i \in C_j^{\mathcal{B}}$). Then, by the monotonicity property, we know that $\mathcal{A} \models \delta(a_1, \ldots, a_k, b_1, \ldots, b_k)$ will hold.

Second, we argue that the distinctness property holds, that is, $a_p \neq a_q$ whenever $\dot{x}_p \in P_i$, $\dot{x}_q \in P_j$, and $i \neq j$. However, our construction ensured that we always have $a_r \in C_s^{\mathcal{B}}$ for the s with $\dot{x}_r \in P_s$. In particular, $\dot{x}_p \in P_i$ and $\dot{x}_q \in P_j$ for $i \neq j$ implies that a_p and a_q lie in two different color classes and are, hence, distinct. □

By the claim, $\mathcal{A} \models \beta$ is equivalent to there being an l-coloring $\mathcal{B}$ of $\mathcal{A}$ such that $\mathcal{B} \models \alpha'$ (β was a block in our main formula of the form $\exists \dot{x}_1 \cdots \exists \dot{x}_k \big(\text{distinct}(P_1, \ldots, P_l) \wedge \alpha\big)$ where there are no weak variables other than the x_i). We now apply Theorem 2 to α' (as ϕ), which yields a new formula α'' (called ϕ' in the theorem) with the property $\mathcal{A} \models \alpha'' \iff \mathcal{A} \models \beta$. The interesting thing about α'' is, of course, that it has the same quantifier rank and the same number of variables as α' plus some constant. Most importantly, we already pointed out earlier that α' does not contain any weak variables and, hence, the quantifier rank of α'' is the same as the strong quantifier rank of β and the number of variables in α'' is the same as the number of strong variables in β—plus some constant.

Applying this transformation to our running example ϕ and choosing as β once more the subformula starting with $\exists \dot{x}_7 \exists \dot{x}_8$, we would get the following formula (ignoring the technical issues how, exactly, the hashing is implemented, see the proof of Theorem 2 for the details):

$$\begin{aligned}
&\forall v \exists \dot{x}_1 \exists \dot{x}_2 \big(\dot{x}_2 \neq \dot{x}_1 \wedge \exists a \exists v_1 (v_1 = \dot{x}_1 \wedge Eav_1) \wedge \\
&\qquad \forall c \big(\exists \dot{x}_3 \exists \dot{x}_4 (\qquad \exists v_1 (v_1 = \dot{x}_3 \wedge \exists v_2 (v_2 = \dot{x}_4 \wedge Ev_1 v_2))) \vee \\
&\qquad\quad \bigvee\nolimits_g \exists p \exists q \exists v_1 (\text{HASH}_g(v_1, p, q) = 1 \wedge \exists v_2 (\text{HASH}_g(v_2, p, q) = 2 \wedge Ev_1 v_2)) \vee \\
&\qquad\quad \exists \dot{x}_5 \exists \dot{x}_6 (\dot{x}_5 \neq \dot{x}_6 \wedge \exists z (Pz \wedge Qcaz))) \big).
\end{aligned}$$

We can now repeat the transformation to replace each block β in this way. Observe that in each transformation we can reuse the variables (in particular, p and q) introduced by the color coding:

$$\begin{aligned}
&\forall v \bigvee\nolimits_g \exists p \exists q \big(\exists a \exists v_1 (\text{HASH}_g(v_1, p, q) = 1 \wedge Eav_1) \wedge \\
&\qquad \forall c \big(\bigvee\nolimits_g \exists p \exists q \exists v_1 (\text{HASH}_g(v_1, p, q) = 1 \wedge \exists v_2 (\text{HASH}_g(v_2, p, q) = 1 \wedge Ev_1 v_2)) \vee \\
&\qquad\quad \bigvee\nolimits_g \exists p \exists q \exists v_1 (\text{HASH}_g(v_1, p, q) = 1 \wedge \exists v_2 (\text{HASH}_g(v_2, p, q) = 2 \wedge Ev_1 v_2)) \vee \\
&\qquad\quad \bigvee\nolimits_g \exists p \exists q \exists z (Pz \wedge Qcaz)) \big).
\end{aligned}$$

In conclusion, we see that we can transform the original formula ϕ to a new formula ϕ' with the following properties:

- We added new variables and quantifiers to ϕ' compared to ϕ during the first transformation steps, but the number we added depended only on the signature τ (it was the maximum arity of relations in τ plus possibly 2).
- We then removed all weak variables from ϕ in ϕ'.
- We added some variables to ϕ' each time we applied Theorem 2 to a block β. The number of variables we added is constant since Theorem 2 adds only a constant number of variables and since we can always reuse the same set of variables each time the theorem is applied.
- We also added some quantifiers to ϕ' each time we applied Theorem 2, which increases the quantifier rank of ϕ' compared to ϕ by more than a constant. However, the essential quantifiers we add are $\exists p \exists q$ and these are always added directly after a universal quantifier or directly after a disjunction after a universal quantifier. Since the strong quantifier rank of ϕ is at least the quantifier rank of ϕ where we only consider the universal quantifiers (the "universal quantifier rank"), the two added

nested quantifiers per universal quantifiers can add to the quantifier rank of ϕ' at most twice the universal quantifier rank.

Putting it all together, we see that ϕ' is equivalent to ϕ, that ϕ' has a quantifier rank that is at most $3\,\text{strong-qr}(\phi) + \text{arity}(\tau) + O(1)$, and the ϕ' contains at most $\text{strong-bound}(\phi) + \text{arity}(\tau) + O(1)$ variables.

This concludes the proof of Theorem 3. □

Since $\text{qr}(\phi) = \text{qr}(\neg\phi)$, a trivial duality argument shows that the above theorem also holds for formulas ϕ without existential weak quantifiers (just apply the theorem to the negation normal form of $\neg\phi$). The more interesting observation is that we can still remove all existential and universal weak quantifiers from a formula when both are present in an arbitrarily complex intertwined form:

Theorem 4. *Let τ be an arithmetic signature. Then for every τ-formula ϕ in negation normal form there is a τ-formula ϕ' such that*

1. $\phi' \equiv \phi$ *(on finite structures),*
2. $\text{qr}(\phi') = 3 \cdot \text{strong-qr}(\phi) + \text{arity}(\tau) + O(1)$, *and*
3. $|\text{bound}(\phi')| = |\text{strong-bound}(\phi)| + \text{arity}(\tau) + O(1)$.

Proof. Given a formula ϕ that contains both existential and universal weak quantifiers, we apply a syntactic preprocessing that "separates these quantifiers and moves them before their dual strong quantifiers." The key observation that makes these transformations possible in the mixed case is that weak existential and weak universal quantifiers commute: For instance, $\exists\dot{x}(\alpha \wedge \forall\dot{y}(\beta)) \equiv \forall\dot{y}(\beta \wedge \exists\dot{x}(\alpha))$ since $\dot{x}$ and $\dot{y}$ cannot depend on one another by the core property of weak quantifiers (α cannot contain $\dot{y}$ and β cannot contain $\dot{x}$). Once we have sufficiently separated the quantifiers, we can repeatedly apply Theorem 3 or its dual to each block individually.

As a running example, let us use the following formula ϕ:

$$\exists\dot{x}\exists a(E\dot{x}a \wedge \forall b(Eba \vee Eab) \wedge \forall\dot{y}(Ea\dot{y} \wedge \exists\dot{z}(Ea\dot{z})) \wedge \exists\dot{w}(E\dot{w}a)),$$

which mixes existential and universal weak variables rather freely.

Similar to the proof of Theorem 3, for technical reasons we first add the superfluous quantifiers $\exists v\forall v$ for a fresh strong variable v at the beginning of the formula.

Our main objective is to get rid of alternations of weak universal and weak existential quantifiers without a strong quantifier in between. In the example, this is the case, for instance, for $\exists\dot{x}(\ldots\forall\dot{y}(\ldots\exists\dot{z}\ldots))$. We get rid of these situations by pushing all quantifiers (weak or strong) down as far as possible (later on, when we apply Theorem 3, we will push them up once more). Let us write $\mathring{x}$ to indicate that x may be a weak or a strong variable.

If β does not contain $\mathring{x}$ as a free variable, we can apply the below four equivalences from left to right (and, of course, commutatively equivalent ones). Note that since the definition of weak variables forbids that a universally bound variable depends on an existential weak variable (and vice versa), the condition "β does not contain $\mathring{x}$" is true, in particular, whenever $\mathring{x}$ is weak and β starts with a universal quantifier in the first two lines or with an existential quantifier in the last two lines.

$$\exists\mathring{x}(\alpha \wedge \beta) \equiv \exists\mathring{x}(\alpha) \wedge \beta, \tag{11}$$
$$\exists\mathring{x}(\alpha \vee \beta) \equiv \exists\mathring{x}(\alpha) \vee \beta, \tag{12}$$
$$\forall\mathring{x}(\alpha \wedge \beta) \equiv \forall\mathring{x}(\alpha) \wedge \beta, \tag{13}$$
$$\forall\mathring{x}(\alpha \vee \beta) \equiv \forall\mathring{x}(\alpha) \vee \beta. \tag{14}$$

Furthermore, we also apply the following general equivalences as long as possible:

$$\exists \dot{x}(\alpha \wedge (\beta \vee \gamma)) \equiv \exists \dot{x}((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)), \tag{15}$$
$$\exists \dot{x}(\alpha \vee \beta) \equiv \exists \dot{x}(\alpha) \vee \exists \dot{x}(\beta), \tag{16}$$
$$\forall \dot{x}(\alpha \vee (\beta \wedge \gamma)) \equiv \forall \dot{x}((\alpha \vee \beta) \wedge (\alpha \vee \gamma)), \tag{17}$$
$$\forall \dot{x}(\alpha \wedge \beta) \equiv \forall \dot{x}(\alpha) \wedge \forall \dot{x}(\beta). \tag{18}$$

Applied to our example, we would get:

$$\exists v \forall v \, \exists \dot{x} \exists a (E\dot{x}a \wedge \forall b (Eba \vee Eab) \wedge \forall \dot{y} (Ea\dot{y}) \wedge \exists \dot{z} (Ea\dot{z}) \wedge \exists \dot{w} (E\dot{w}a)).$$

As a final transformation, we "sort" the operands of disjunctions and conjunctions: We replace a subformula $\alpha \wedge \beta$ in ϕ by $\beta \wedge \alpha$ and we replace $\alpha \vee \beta$ by $\beta \vee \alpha$, whenever β contains no weak universal variables, but α does, and also whenever α contains no weak existential variables, but β does. For our example, this means that we get the following:

$$\exists v \forall v \, \exists \dot{x} \exists a (\exists \dot{z} (Ea\dot{z}) \wedge \exists \dot{w} (E\dot{w}a) \wedge E\dot{x}a \wedge \forall b (Eba \vee Eab) \wedge \forall \dot{y} (Ea\dot{y})).$$

The purpose of the transformations was to achieve the situation described in the next claim:

Claim 3. *Assume that the above transformations have been applied exhaustively to ϕ and assume ϕ contains both existential and universal weak variables. Consider the maximal subformulas α_i of ϕ that contain no weak universal variables and the maximal subformulas β_i of ϕ that contain no weak existential variables. Then for some i and some γ one of the following formulas is a subformula of ϕ: $\forall x(\alpha_i \vee \gamma)$ or $\exists x(\gamma \wedge \beta_i)$.*

In our example, there is only a single maximal α_1, namely $\exists \dot{z}(Ea\dot{z}) \wedge \exists \dot{w}(E\dot{w}a) \wedge E\dot{x}a \wedge \forall b(Eba \vee Eab)$, and a single maximal β_1, namely $\forall b(Eba \vee Eab) \wedge \forall \dot{y}(Ea\dot{y})$. The claim holds since $\exists a(\gamma \wedge \beta_1)$ is a subformula for $\gamma = \exists \dot{z}(Ea\dot{z}) \wedge \exists \dot{w}(E\dot{w}a) \wedge E\dot{x}a$.

Proof. Consider any α among the α_i. Since α is maximal but α is not all of ϕ, there must be a β among the β_i such that either $\alpha \vee \beta$ or $\alpha \wedge \beta$ is also a subformula of ϕ. Let us call it δ and consider the minimal subformula η of ϕ that contains δ and starts with a quantifier.

This quantifier cannot be a weak quantifier: Suppose it is $\exists \dot{x}$ (the case $\forall \dot{x}$ is perfectly symmetric). Since we can no longer apply one of the equivalences (11)–(18), the formula η must have the form $\exists \dot{x} \bigwedge_i \psi_i$ (where the ψ_i are not of the form $\rho \wedge \sigma$) such that all ψ_i contain $\dot{x}$ (otherwise (11) would be applicable) and such that none of the ψ_i is of the form $\rho \vee \sigma$ (otherwise (15) would be applicable). This implies that all ψ_i start with a quantifier. Since η was minimal to contain δ, we conclude that one ψ_i must be α and another one must be β. Then, β contains a weak existential variable, namely $\dot{x}$, which we ruled out.

Since η does not start with a weak quantifier, it must start with a strong quantifier. If it is $\exists x$, by the same argument as before we get that η must have the form $\exists x \bigwedge_i \psi_i$ with some ψ_i equal to α and some other ψ_j equal to β. Then, we have found the desired subformula of ϕ if we set γ to $\bigwedge_{i \neq j} \psi_i$. If the strong quantifier is $\forall x$, a perfectly symmetric argument shows that η must have the form $\forall x \bigvee_i \psi_i$ with some $\psi_j = \alpha$, which implies the claim for $\gamma = \bigvee_{i \neq j} \psi_i$. □

The importance of the claim for our argument is the following: As long as ϕ still contains both existential and universal weak variables, we still find a subformula α or β that contains only existential or universal weak variables such that if we go up from this subformula in the syntax tree of ϕ, the next quantifier we meet is a strong quantifier. This means that we can now apply Theorem 3 or its dual to this subformula, getting an equivalent new formula α' or β' whose quantifier rank equals the strong quantifier rank of α or β, respectively, times a constant factor. Furthermore, similar to the argument at the end of the proof of Theorem 3 where we processed one β after another, each time a replacement takes place, there is a strong quantifier that contributes to the strong quantifier rank of ϕ.

This concludes the proof of Theorem 4. □

4. Syntactic Proofs and Natural Problems

The special allure of descriptive complexity theory lies in the possibility of proving that a problem has a certain complexity just by describing the problem in the right way. The "right way" is, of course, a logical description that has a certain syntax (such as having a bounded strong quantifier rank). In the following we present such descriptions for several natural problems and thereby bound their complexity "in a purely syntactic way." First, however, we present "syntactic tools" for describing problems more easily. These tools are built on top of the notion of strong and weak quantifiers.

4.1. Syntactic Tools: New Operators

It is common in mathematical logic to distinguish between the core syntax and additional "shorthands" built on top of the core syntax. For instance, while $\neg$ and $\vee$ are typically considered to be part of the core syntax of propositional logic, the notation $a \to b$ is often seen as a shorthand for $\neg a \vee b$. In a similar way, we now consider the notions of weak variables and quantifiers introduced in the previous section as our "core syntax" and build a number of useful shorthands on top of them. Of course, just as $a \to b$ has an intended semantic meaning that the expansion $\neg a \vee b$ of the shorthand must reflect, the shorthands we introduce also have an intended semantic meaning, which we specify.

As a first example, consider the common notation $\exists^{\geq k} x(\phi(x))$, whose intended semantics is "there are at least k different elements in the universe that make $\phi(x)$ true." This notation is often considered as a shorthand for $\exists x_1 \cdots \exists x_k \bigwedge_{i \neq j} x_i \neq x_j \wedge \bigwedge_{i=1}^{k} \phi(x_i)$, but we will consider it a shorthand for the equivalent, but slightly more complicated formula $\exists \dot{x}_1 \cdots \exists \dot{x}_k \bigwedge_{i \neq j} \dot{x}_i \neq \dot{x}_j \wedge \bigwedge_{i=1}^{k} \exists x(x = \dot{x}_i \wedge \phi(x))$. The difference is, of course, that the strong quantifier rank is now much lower and, hence, by Theorem 3 we can replace any occurrence of $\exists^{\geq k} x(\phi(x))$ by a formula of quantifier rank $\mathrm{qr}(\phi) + O(1)$. In all of the following notations, k and s are arbitrary values. The indicated strong quantifier rank for the notation is that of its expansion. The semantics describe which structures $\mathcal{A}$ are models of the formula.

Notation ($\exists^{\geq k} x(\phi(x))$). **Strong-qr:** $1 + \text{strong-qr}(\phi)$
Semantics There are k distinct $a_1, \ldots, a_k \in |\mathcal{A}|$ with $\mathcal{A} \models \phi(a_i)$ for all i.
Expansion $\exists \dot{x}_1 \cdots \exists \dot{x}_k \bigwedge_{i \neq j} \dot{x}_i \neq \dot{x}_j \wedge \bigwedge_{i=1}^{k} \exists x(x = \dot{x}_i \wedge \phi(x))$

Notation ($\exists^{\leq k} x(\phi(x))$). **Strong-qr:** $1 + \text{strong-qr}(\phi)$
Semantics There are at most k distinct $a_1, \ldots, a_k \in |\mathcal{A}|$ with $\mathcal{A} \models \phi(a_i)$ for all i.
Expansion $\forall \dot{x}_1 \cdots \forall \dot{x}_{k+1} \bigvee_{i \neq j} \dot{x}_i = \dot{x}_j \vee \bigvee_{i=1}^{k+1} \forall x(x \neq \dot{x}_i \vee \neg\phi(x))$ $(\equiv \neg\exists^{\geq k+1} x(\phi(x)))$

Notation ($\exists^{=k} x(\phi(x))$). **Strong-qr:** $1 + \text{strong-qr}(\phi)$
Semantics There are exactly k distinct $a_1, \ldots, a_k \in |\mathcal{A}|$ with $\mathcal{A} \models \phi(a_i)$ for all i.
Expansion $\exists^{\geq k} x(\phi(x)) \wedge \exists^{\leq k} x(\phi(x))$

The next notation is useful for "binding" a set of vertices to weak or strong variables. The binding contains the allowed "single use" of the weak variables in the sense of Definition 3, but they can still be used in inequality literals. Let $\mathring{x}$ indicate that x may be weak or strong.

Notation ($\{\mathring{x}_1, \ldots, \mathring{x}_k\} = \{x \mid \phi(x)\}$). **Strong-qr:** $1 + \text{strong-qr}(\phi)$
Semantics Let $a_1, \ldots, a_k \in |\mathcal{A}|$ be the assignments to the $\mathring{x}_i$ (note that they need not be distinct). Then $\{a_1, \ldots, a_k\} = \{a \in |\mathcal{A}| \mid \mathcal{A} \models \phi(a)\}$ must hold.
Expansion

$$\begin{aligned}
&\textstyle\bigwedge_{i=1}^{k} \exists x\big(x = \mathring{x}_i \wedge \phi(x)\big) \wedge && \text{// ensure } \{\mathring{x}_1, \ldots, \mathring{x}_k\} \subseteq \{x \mid \phi(x)\} \\
&\textstyle\bigvee_{s=1}^{k} \big(\exists^{=s} x(\phi(x)) \wedge && \text{// bind } s \text{ to } |\{x \mid \phi(x)\}| \\
&\quad \textstyle\bigvee_{I \subseteq \{1,\ldots,k\}, |I|=s} \bigwedge_{i,j \in I, i \neq j} \mathring{x}_i \neq \mathring{x}_j\big). && \text{// ensure } |\{\mathring{x}_1, \ldots, \mathring{x}_k\}| \geq s
\end{aligned}$$

(We remark that in the above notation, the last line does, indeed, by itself only ensure that $|\{\mathring{x}_1, \dots, \mathring{x}_k\}| \geq s$ holds. However, in conjunction with the two lines before it, we get that $|\{\mathring{x}_1, \dots, \mathring{x}_k\}| = s$ must hold for the formula to be true.)

The final notation can be thought of as a "generalization of $\exists^{=k}$" where we not only ask whether there are exactly k distinct a_i with a property ϕ, but whether these a_i then also have an arbitrary special additional property. Formally, let $Q \subseteq \text{STRUC}[\tau]$ be an arbitrary τ-problem. We write $\mathcal{A}[I]$ for the substructure of $\mathcal{A}$ induced on a subset $I \subseteq |\mathcal{A}|$.

Notation $(\text{INDUCED}^{\text{size}=k}\{x \mid \phi(x)\} \in Q)$. **Strong-qr:** $1 + \text{strong-qr}(\phi) + \text{arity}(\tau)$
Semantics The set $I = \{a \in |\mathcal{A}| \mid \mathcal{A} \models \phi(a)\}$ has size exactly k and $\mathcal{A}[I] \in Q$.
Expansion Assuming for simplicity that τ contains only E^2 as non-arithmetic predicate:

$$\exists^{=k} x(\phi(x)) \wedge \bigvee_{\mathcal{A} \in Q, |\mathcal{A}| = \{1,\dots,k\}} \bigwedge_{(i,j) \in E^{\mathcal{A}}} \exists x \exists y(\pi_i(x) \wedge \pi_j(y) \wedge Exy) \wedge \bigwedge_{(i,j) \notin E^{\mathcal{A}}} \exists x \exists y(\pi_i(x) \wedge \pi_j(y) \wedge \neg Exy),$$

where $\pi_i(x)$ is a shorthand for $\phi(x) \wedge \exists^{=i-1} z(z < x \wedge \phi(z))$, which binds x to the ith element of the universe with property ϕ.

Notation $(\text{INDUCED}^{\text{size}\leq k}\{x \mid \phi(x)\} \in Q)$. **Strong-qr:** $1 + \text{strong-qr}(\phi) + \text{arity}(\tau)$
Semantics The set $I = \{a \in |\mathcal{A}| \mid \mathcal{A} \models \phi(a)\}$ has size at most k and $\mathcal{A}[I] \in Q$.
Expansion $\bigvee_{s=0}^{k} \text{INDUCED}^{\text{size}=s}\{x \mid \phi(x)\} \in Q$

4.2. Bounded Strong-Rank Description of Vertex Cover

The first advanced problem for which we now use our framework to "prove syntactically" that it lies in para-AC^0, is the vertex cover problem. A vertex cover of an undirected graph $G = (V, E)$ is a subset $X \subseteq V$ with $e \cap X \neq \emptyset$ for all $e \in E$. The problem p_k-VERTEX-COVER asks whether a graph has a cover X with $|X| \leq k$. The most natural way to describe the problem is in terms of the following family $(\psi_k)_{k \in \mathbb{N}}$:

$$\psi_k = \exists x_1 \cdots \exists x_k \forall u \forall v \big(Euv \to \bigvee_i (x_i = u \vee x_i = v)\big).$$

However, the strong quantifier rank of this family is clearly not bounded since all quantifiers are strong (all x_i depend on the universally bound variables u and v). Thus, we need a different way of describing the vertex cover problem:

Theorem 5 ([4,7]). p_k-VERTEX-COVER $\in$ para-AC^0.

Proof. We describe the problem using a family $(\phi_k)_{k \in \mathbb{N}}$ of constant strong quantifier rank that expresses the well-known Buss kernelization "using logic": Let $\text{HIGH}(x) = \exists^{\geq k+1} y\,(Exy)$ expresses that x is a high-degree vertex. Buss observed that all high-degree vertices must be part of a vertex cover of size at most k. Thus, $h \leq k$ must hold for the unique h with $\exists^{=h} x(\text{HIGH}(x))$. A remaining vertex is interesting if it is connected to at least one non-high-degree vertex: $\text{INTERESTING}(x) = \neg\,\text{HIGH}(x) \wedge \exists y(Exy \wedge \neg\,\text{HIGH}(y))$. If there are more than $(k-h)(k+1) \leq k^2 + k$ interesting vertices, there cannot be a size-k vertex cover—and if there are less, the original graph can only have a size-k vertex cover if the graph induced on the interesting vertices has a vertex cover of size $k - h$. In symbols: $\phi_k = \bigvee_{h=0}^{k} \big(\exists^{=h} x(\text{HIGH}(x)) \wedge \text{INDUCED}^{\text{size}\leq k^2+k}\{x \mid \text{INTERESTING}(x)\} \in Q_{k-h}\big)$ for the problem $Q_s = \{\mathcal{G} \mid \mathcal{G} \text{ has a vertex cover of size } s\}$ (recall that for the INDUCED notation we allow arbitrary problems). □

While the family (ϕ_k) constructed in the above proof is not extremely complex and much simpler than formulas or circuits found in the literature [4,7] for proving p_k-VERTEX-COVER $\in$ para-AC^0 without using our framework, the family certainly lacks the elegance of the earlier "natural" family (ψ_k). It seems like an interesting goal to find new syntactic properties that broaden the notion of weak quantifiers so that the existential quantifiers in ψ_k become weak. We believe that this is possible, but also point out that

while any such properties must apply to the quantifiers in the ψ_k, they may not apply to the almost identical formulas

$$\psi'_k = \exists x_1 \cdots \exists x_k \forall u \forall v \big(\bigvee_{i \neq j} (x_i = u \wedge x_j = v) \rightarrow Euv \big), \tag{19}$$

which describe the parameterized clique problem—a problem provably not in para-AC^0.

4.3. Bounded Strong-Rank Description of Hitting Set

Hitting sets generalize vertex covers to hypergraphs, which are pairs (V, E) where the members of E are called hyperedges and we have $e \subseteq V$ for all $e \in E$. Hitting sets are still sets $X \subseteq V$ with $e \cap X \neq \emptyset$ for all $e \in E$. The problem $\mathrm{p}_{k,d}$-HITTING-SET asks whether a hypergraph with $\max_{e \in E} |e| \leq d$ has a hitting set X with $|X| \leq k$. Formally, let $d(H)$ be the maximum size of any hyperedge in H and $\mathrm{p}_{k,d}$-HITTING-SET be the set of all pairs $(\mathcal{H}, \max(k, d))$ such that $\mathcal{H}$ encodes (see below for details) a hypergraph H with $d(H) \leq d$ and for which there is a hitting set of size at most k. The problem p_k-VERTEX-COVER is basically this problem restricted to the parameter value $d = 2$ (if one models size-1 hyperedges e as self-loops and ignores the case of the empty hyperedge, which can never be hit by any set X).

While by no means trivial, it is not too hard to generalize the arguments used in Theorem 5 to prove that the parameterized hitting sets problem lies in para-AC^0 when d is fixed and k is the parameter. It is much harder to prove the following result, where both d and k are parameters:

Theorem 6 ([19]). $\mathrm{p}_{k,d}$-HITTING-SET $\in$ para-AC^0.

Before we prove the theorem using our framework, we need to fix how we model hypergraphs (V, E) as logical structures. We use a τ_{hyper}-structure $\mathcal{H}$ for $\tau_{\mathrm{hyper}} = (\mathrm{VERTEX}^1, \mathrm{HYPEREDGE}^1, \mathrm{IN}^2)$. Let $|\mathcal{H}| = V \cup E$, $\mathrm{VERTEX}^{\mathcal{H}} = V$, set $\mathrm{HYPEREDGE}^{\mathcal{H}} = E$, and $\mathrm{IN}^{\mathcal{H}} = \{(v, e) \mid v \in e \in E\}$. The other way round, given a τ_{hyper}-structure $\mathcal{H}$, we consider it as the following hypergraph $H(\mathcal{H}) = (V, E)$ with $V = \mathrm{VERTEX}^{\mathcal{H}}$ and $E = \{\mathrm{set}(e) \mid e \in \mathrm{HYPEREDGE}^{\mathcal{H}}\}$, where we write $\mathrm{set}(e)$ for $\{v \mid (v, e) \in \mathrm{IN}^{\mathcal{H}}\}$.

Note that we allow the universe of $\mathcal{H}$ to contain elements e that are neither vertices nor hyperedge-representing elements, but their $\mathrm{set}(e)$ do not contribute to E. We also allow that two different elements $e, e' \in |\mathcal{H}|$ represent the same set $\mathrm{set}(e) = \mathrm{set}(e')$. This can be problematic in a kernelization: When we identify a kernel set E' of hyperedges, there could still be a large (non-parameter-dependent) number of elements in the universe that represent these hyperedges—meaning that these elements do not form a kernel themselves. Fortunately, this can be fixed: We can easily check whether two elements represent the same set using $\forall x(\mathrm{IN}\, xe \leftrightarrow \mathrm{IN}\, xe')$ and then always consider only the first representing element with respect to the ordering $<$ of the universe. For this reason, we will assume in the following that for any subset $s \subseteq V$ there is at most one $e \in |\mathcal{H}|$ with $s = \mathrm{set}(e)$.

Proof of Theorem 6. The idea behind the proof is a (very strong) generalization of the Buss kernel argument from the proof of Theorem 5. As in that proof, we will present a family $(\phi_{k,d})_{k,d \in \mathbb{N}}$ of bounded strong quantifier rank that describes $\mathrm{p}_{k,d}$-HITTING-SET. First, there are two simple preliminaries: Testing whether $d(\mathcal{H}) \leq d$ holds is easy to achieve using $\forall e(\mathrm{HYPEREDGE}\, e \rightarrow \exists^{\leq d} v(\mathrm{IN}\, ve))$, so let us assume that this is the case and let us write $H = (V, E)$ for $H(\mathcal{H})$. Furthermore, let us write $\mathrm{SUBSET}\, ef$ for $\forall x(\mathrm{IN}\, xe \rightarrow \mathrm{IN}\, xf)$, which indicates that $\mathrm{set}(e) \subseteq \mathrm{set}(f)$.

Representing subsets of hyperedges: recall that the core idea of the kernelization of the vertex cover problem is that a "high-degree vertex" must be part of a vertex cover. Rephrased in the language of hypergraphs, a graph is a hypergraph H with $d(H) = 2$, a vertex cover is a hitting set, and making a high-degree vertex v part of a hitting set is

(in essence) the same as removing all edges containing v and then adding the singleton hyperedge $\{v\}$, which can clearly only be hit by making v part of the hitting set.

In the general case, we will also remove hyperedges from the hypergraph and replace them by smaller hyperedges (though, no longer, by singletons) and we will do so repeatedly. The problem is that adding hyperedges is difficult in our encoding since this means that we would have to add elements to the universe of the logical structure that represent the new hyperedges. Although these problems can be circumvented by complex syntactic trickery, we feel it is cleaner to do the following: We reduce the original hitting set problem to a new version, where the universe already contains all the necessary elements for representing the hyperedges we might wish to add later on.

In detail, we define a subset $\mathrm{p}_{k,d}$-HITTING-SET$' \subseteq \mathrm{p}_{k,d}$-HITTING-SET as follows: It contains only those $(\mathcal{H}, \max(k,d))$ such that for every $e \in \text{HYPEREDGE}^{\mathcal{H}}$ and every subset $s \subseteq \text{set}(e)$ there is an $e' \in |\mathcal{H}|$ with $s = \text{set}(e')$. In other words, for every subset s of any hyperedge there must already be an element e "in store" in the universe that represents it.

We can reduce $\mathrm{p}_{k,d}$-HITTING-SET to $\mathrm{p}_{k,d}$-HITTING-SET$'$ by adding for an input $\mathcal{H}$, if necessary, elements to the universe that represent all these subsets (note that the set $\text{HYPEREDGE}^{\mathcal{H}}$ is not changed in the reduction, we just add elements to the universe for "potential, future" hyperedges). We are helped by the fact that we have an upper bound d on the size of the hyperedges, which means that the maximum blowup of the universe in this reduction is by the parameter-dependent value of 2^d. This reduction can be realized via a bounded rank reduction (recall Definition 1):

Claim 4. $\mathrm{p}_{k,d}$-HITTING-SET $\leq_{\mathrm{br}} \mathrm{p}_{k,d}$-HITTING-SET$'$.

Proof. In the reduction, we do not change the parameter, so we set $\iota_{\max(k,d),\max(k,d)} = \top$ and $\iota_{x,x'} = \bot$ otherwise. For the first-order queries, we wish to map a hypergraph $\mathcal{H}$ to a new version $\mathcal{H}'$ in which for every subset of a hyperedge there is already an element in the universe representing this subset. This means that the size of the universe can increase from $|\mathcal{H}|$ to at most $2^d|\mathcal{H}|$. (If there is a hyperedge of size larger than d in the input, we can yield a trivial "no" instance as output.) We use a first-order query of width 2, meaning that the universe size gets enlarged from $|\mathcal{H}|$ to $|\mathcal{H}|^2$. This will be larger than $2^d|\mathcal{H}|$ for all sufficiently large universes. Since, with respect to $f_{\max(k,d)}$ the number 2^d is a constant, we can apply Lemma 1 to take care of those inputs whose universes are smaller than 2^d and directly map them to the correct instances. For the large instances, we now have a universe that is "large enough" to contain an element for each subset of a hyperedge and it is not difficult (but technical) to use the bit predicate to define the correct predicates HYPEREDGE, VERTEX, and IN in terms of the original structure. □

Thus, in the following, we may assume that for every hyperedge in the input structure for all subsets of this hyperedge we already have an element in the universe representing this subset.

Finding sunflowers: we first show a way of kernelizing the hitting set problem, due to Chen et al. [4], that "almost works." The core idea is to detect and collapse sunflowers in the input hypergraph [20]. A sunflower of size $k+1$ with core c is a set $\{p_1, \ldots, p_{k+1}\} \subseteq E$ of distinct hyperedges, called petals, such that for all $i \neq j$ we have $p_i \cap p_j = c$. In other words, all petals contain the core but are otherwise pairwise distinct. For convenience, we also assume that all petals are proper supersets of the core. The important observation is that if a sunflower of size $k+1$ has a hitting set of size k, then the core must also be hit—and when the core is hit, all petals are hit. This means that we can just replace a sunflower by its core when we are looking for size-k hitting sets.

The following formula tests whether $\text{set}(c)$ is the core of a sunflower of size $k+1$:

$$\text{CORE}\, c = \exists \dot{p}_1^1 \cdots \exists \dot{p}_{k+1}^d \bigwedge_{i \neq j} \bigwedge_{r,s \in \{1,\ldots,d\}} \dot{p}_i^r \neq \dot{p}_j^s \wedge \tag{20}$$

$$\bigwedge_{i=1}^{k+1} \exists e \big(\text{HYPEREDGE}\, e \wedge \text{SUBSET}\, ce \wedge \{\dot{p}_i^1, \ldots, \dot{p}_i^d\} = \{v \mid \text{IN}\, ve \wedge \neg\, \text{IN}\, vc\} \big). \tag{21}$$

Here, (20) guarantees that the petals are pairwise disjoint outside the core and (21) checks that the petals are supersets of c and when we add p_i^1 to p_i^d (which are not necessarily distinct) to c, we get a present hyperedge.

The "collapsing" of sunflowers to their cores can now be done as follows: We define a formula with e as a free variable that is true when $\mathrm{set}(e)$ is a core or when $\mathrm{set}(e)$ is not a superset of any core (otherwise, we need not include $\mathrm{set}(e)$ since we include the core of a sunflower that contains it, instead):

$$\textsc{core}\, e \vee (\textsc{hyperedge}\, e \wedge \neg\exists c(\textsc{core}\, c \wedge \textsc{subset}\, ce)). \tag{22}$$

The importance of the above formula lies in the following fact: The number of hyperedges for which the second part of the formula is true (that is, which are not supersets of a core of a sunflower of size $k+1$), is bounded by a function in k and d. This is due to the famous Sunflower Lemma [20] which states that if a hypergraph has more than $k^d d!$ hyperedges, it contains a sunflower of size $k+1$ (which has a core).

This means that if $\textsc{core}\, e$ were to hold for just a few hyperedges, (22) would describe a kernel for the hitting set problem, meaning that the number of hyperedges remaining would be bounded by a purely parameter-dependent value. We could then proceed as in the proof of Theorem 5: In order to determine whether the original hypergraph has a hitting set of size k, we only need to check whether the hypergraph induced on the remaining hyperedges has such a hitting set—and since a kernel has a purely parameter-dependent size, we could use the $\textsc{induced}$ notation to solve the hitting set problem on the vertices and hyperedges for which (22) holds. Unfortunately, it is possible to construct hypergraphs such that $\textsc{core}\, e$ still holds for a very large (not only parameter-dependent) number of hyperedges.

However, we know that a core always has a smaller size than any petal in its sunflower. In particular, all cores have maximum size $d-1$. Thus, if we "view $\textsc{core}$ as our new $\textsc{hyperedge}$ predicate," we get "cores of cores":

$$\begin{aligned}\textsc{core}^2\, c = \exists \dot{p}_1^1 \cdots \exists \dot{p}_{k+1}^d \textstyle\bigwedge_{i\neq j} \bigwedge_{r,s\in\{1,\ldots,d\}} \dot{p}_i^r \neq \dot{p}_j^s \wedge \\ \textstyle\bigwedge_{i=1}^{k+1} \exists e' \big(\textsc{core}\, e' \wedge \textsc{subset}\, ce' \wedge \\ \{\dot{p}_i^1, \ldots, \dot{p}_i^d\} = \{v \mid \textsc{in}\, ve' \wedge \neg\, \textsc{in}\, vc\}\big).\end{aligned}$$

Note that $\text{strong-qr}(\textsc{core}^2\, c) = \text{strong-qr}(\textsc{core}\, c) + 1 = 2$ since we had to add a new strong quantifier ($\exists e'$) whose scope contains $\textsc{core}\, e'$, which adds its own strong quantifier ($\exists e$).

By the same argument as earlier, we get that the number of e for which the following formula holds equals the number of cores of cores plus something that only depends on the parameters k and d:

$$\begin{aligned}\textsc{core}^2\, e \vee (\textsc{core}\, e \wedge \neg\exists c(\textsc{core}^2\, c \wedge \textsc{subset}\, ce)) \\ \vee (\textsc{hyperedge}\, e \wedge \neg\exists c(\textsc{core}\, c \wedge \textsc{subset}\, ce)).\end{aligned}$$

Still, the number of cores of cores can be large, but they all have size at most $d-2$. Repeating the argument a further $d-2$ times, we finally get the predicate $\textsc{kernel}\, e$:

$$\textsc{core}^d\, e \vee \textstyle\bigvee_{i=1}^{d} (\textsc{core}^{i-1}\, e \wedge \neg\exists c(\textsc{core}^i\, c \wedge \textsc{subset}\, ce)), \tag{23}$$

where $\textsc{core}^0$ is of course $\textsc{hyperedge}$ and $\textsc{core}^d\, e$ can only be true for the (sole) e representing the empty set (in which case, there is no hitting set).

Unfortunately, the strong quantifier rank of $\textsc{core}^d$ is d since the definition of $\textsc{core}^i$ in terms of $\textsc{core}^{i-1}$ always adds one strong quantifier nesting (through a new $\exists e'^{\cdots\prime}$). Thus, (23) also has a strong quantifier rank of d while we need $O(1)$.

Finding pseudo-sunflowers: at this point, we need a way of describing cores of cores of cores and so on using a bounded strong quantifier rank. The idea how this can be done was presented in [19], where the notions of pseudo-cores and pseudo-sunflowers are introduced. The definitions are somewhat technical, see below, but the interesting fact about these definitions is that they can be expressed very nicely in a way similar to (20) and (21).

For a level L and a number k, let T_L^k denote the rooted tree in which all leaves are at the same depth L and all inner nodes have exactly $k+1$ children. The root of T_L^k will always be called r in the following. Thus, T_1^k is just a star consisting of r and its $k+1$ children, while in T_2^k each of the $k+1$ children of r has $k+1$ new children, leading to $(k+1)^2$ leaves in total. For each $l \in \text{leaves}(T_L^k) = \{l \mid l \text{ is a leaf of } T_L^k\}$ there is a unique path $(l^0, l^1, \ldots, l^L)$ from $l^0 = r$ to $l^L = l$.

Definition 4 (Pseudo-Sunflowers and Pseudo-Cores, [19]). *Let $H = (V, E)$ be a hypergraph and let L and k be fixed. A set $c \subseteq V$ is called a k-pseudo-core of level L in H if there exists a mapping $S\colon \text{leaves}(T_L^k) \times \{0, 1, \ldots, L\} \to \{e \mid e \subseteq V\}$, called a T_L^k-pseudo-sunflower for H with pseudo-core c, such that for all $l, m \in \text{leaves}(T_L^k)$ with $l \neq m$ we have:*

1. $S(l, 0) = c$.
2. $S(l, 0) \cup S(l, 1) \cup \cdots \cup S(l, L) \in E$.
3. $S(l, i) \cap S(l, j) = \emptyset$ *for* $0 \leq i < j \leq L$, *but* $S(l, i) \neq \emptyset$ *for* $i \in \{1, \ldots, L\}$.
4. *Let $z \in \{1, \ldots, L\}$ be the smallest number such that $l^z \neq m^z$, that is, z is the depth where the path from r to l and the path from r to m diverge for the first time. Then $S(l, z) \cap S(m, z) = \emptyset$ must hold.*

This definition translates almost directly into a formula $\text{PSEUDOCORE}^L\, c$, which starts with a block of weak existential quantifiers, one for each element of $\text{leaves}(T_L^k) \times \{1, \ldots, L\} \times \{1, \ldots, d\}$:

$$
\begin{aligned}
&(\exists \dot{x}_{l,i}^j)_{l \in \text{leaves}(T_L^k), i \in \{1,\ldots,L\}, j \in \{1,\ldots,d\}} \\
&\quad \bigwedge_{l,m \in \text{leaves}(T_L^k), l \neq m, z \text{ as in the definition}} \Big(\\
&\qquad \exists e \big(\text{HYPEREDGE}\, e \wedge \text{SUBSET}\, ce \wedge \{\dot{x}_{l,1}^1, \ldots, \dot{x}_{l,L}^d\} = \{v \mid \text{IN}\, ve \wedge \neg\, \text{IN}\, vc\} \big) \wedge && (24) \\
&\qquad \textstyle\bigwedge_{i \neq j} \bigwedge_{p,q \in \{1,\ldots,d\}} \dot{x}_{l,i}^p \neq \dot{x}_{l,j}^q \wedge && (25) \\
&\qquad \textstyle\bigwedge_{p,q \in \{1,\ldots,d\}} \dot{x}_{l,z}^p \neq \dot{x}_{m,z}^q \Big). && (26)
\end{aligned}
$$

Here, (24) ensures, similarly to (21) for normal sunflowers, that $S(l, 0) \cup S(l, 1) \cup \cdots \cup S(l, L)$ is a hyperedge, item 2 of the definition. The inequalities (25) ensure that item 3 of the definition holds, while (26) ensures item 4.

The important observation is that PSEUDOCORE^L has a strong quantifier rank that is independent of L. Since, as shown in [19], we can use PSEUDOCORE^L as a replacement for CORE^L in (23), we get that the hitting set problem can be described by a family of formulas of constant strong quantifier rank.

This concludes the proof of Theorem 6, which compared to the original proof in [19] is considerably simpler and shorter because of the use of the framework from the present paper. □

4.4. Bounded Strong-Rank Description of Model Checking for First-Order Logic

An important result by Flum and Grohe [10] states that the model checking problem for first-order logic lies in FPT on structures whose Gaifman graph has bounded degree (the Gaifman graph of a logical structure has the structure's universe as its vertex set and has an undirected edge between two vertices u and v if u and v are simultaneously part of some tuple of some relation of the structure; in particular, the Gaifman graph of a directed

graph is its underlying undirected graph). Once more, this result can now be obtained "syntactically." For simplicity, we only consider graphs and let

$$\mathrm{p}_{\psi,\delta}\text{-MC}(\mathrm{FO}) = \{(\mathcal{G}, \mathrm{num}(\psi,\delta)) \mid \mathcal{G} \in \mathrm{STRUC}[(E^2)],\ \psi \in \mathrm{FO},\ \mathcal{G} \models \psi, \text{max-degree}(\mathcal{G}) \leq \delta\}.$$

Theorem 7 ([7,10]). $\mathrm{p}_{\psi,\delta}\text{-MC}(\mathrm{FO}) \in \text{para-AC}^{0\uparrow}$.

Proof. We present a family $(\phi_{\psi,\delta})_{\psi\in\mathrm{FO},\delta\in\mathbb{N}}$ with a bound on the number of strong variables that describes $\mathrm{p}_{\psi,\delta}\text{-MC}(\mathrm{FO})$. Fix ψ and δ. Recall that we fixed the signature for ψ to just $\tau = (E^2)$ for simplicity and, thus, τ-structures are just graphs $\mathcal{G}$. In particular, there are no arithmetic predicates available to ψ (one could, of course, also consider them, but then the Gaifman graph would always be a clique and the claim of the theorem would be boring). In contrast, the $\phi_{\psi,\delta}$ are normal $\mathrm{FO}[+,\times]$ formulas and they have access to arithmetics.

For a graph $\mathcal{G}$ let us write $\bar{\mathcal{G}}$ for the underlying undirected graph and let us write $\bar{E}xy$ as a shorthand for $Exy \vee Eyx$. The first thing we check is that the maximum degree of the input graph $\bar{\mathcal{G}}$ is, indeed, δ. This is rather easy: $\forall x \exists^{\leq\delta} y(\bar{E}xy)$.

For the hard part of determining whether $\mathcal{G} \models \psi$, let $\bar{d}(a,b)$ denote the distance of two vertices in $\bar{\mathcal{G}}$ and let $N_r(a) = \{b \in |\mathcal{G}| \mid \bar{d}(a,b) \leq r\}$ be the ball around a of radius r in $\bar{\mathcal{G}}$. Let $\mathcal{G}[N_r(a)]$ denote the subgraph of $\mathcal{G}$ induced on $N_r(a)$. By Gaifman's Theorem [21] we can rewrite ψ as a Boolean combination of formulas of the following form:

$$\exists x_1 \cdots \exists x_k \Big(\textstyle\bigwedge_{i\neq j} \gamma_{\bar{d}(x_i,x_j)>2r} \wedge \tag{27}$$

$$\textstyle\bigwedge_i \rho(x_i) \Big) \tag{28}$$

where $\gamma_{\bar{d}(x_i,x_j)>2r}$ expresses that $\bar{d}(x_i,x_j) > 2r$ should hold and ρ is r-local, meaning that for all $a \in |\mathcal{G}|$ we have $\mathcal{G} \models \rho(a) \iff \mathcal{G}[N_r(a)] \models \rho(a)$ (the minimum number r for which is the case is called the locality rank of ρ).

We now wish to express the above formula using only a constant number of strong variables. The problem is, of course, that the x_i are not (yet) weak since they are used many times. We fix this in two steps. First, let us tackle (27): Clearly, the x_i will have a pairwise distance of at least $2r$, if the balls of radius r that surround them are pairwise disjoint. Now, because of the bounded degree of the graph, a ball of radius r can have maximum size δ^r. This allows us to bind all members of each ball and testing disjointness is exactly what weak variables are all about.

In detail, let $\gamma_{\bar{d}(x,y)\leq r}$ be the standard formula with two bound variables expressing that there is path from x to y of length at most r in $\bar{\mathcal{G}}$. Then we can express (27) as follows:

$$\exists \dot{x}_1 \cdots \exists \dot{x}_k \exists \dot{y}_1^1 \cdots \exists \dot{y}_k^{\delta^r} \big(\textstyle\bigwedge_{i\neq j} \bigwedge_{p,q\in\{1,\ldots,\delta^r\}} \dot{y}_i^p \neq \dot{y}_j^q \wedge \bigwedge_{i=1}^k \exists x(x = \dot{x}_i \wedge \{\dot{y}_i^1, \ldots, \dot{y}_i^{\delta^r}\} = \{y \mid \gamma_{\bar{d}(x,y)\leq r}\})\big).$$

In the formula, at the end we bind the variables $\dot{y}_i^1, \ldots, \dot{y}_i^{\delta^r}$ exactly to the elements of the ball around $\dot{x}_i$ or radius r; and in the first part we require that all these balls are pairwise disjoint. Note that we do not require all $\dot{y}_i^p$ to be different: If the size of a ball is less than δ^r, we must allow some $\dot{y}_i^p$ and $\dot{y}_i^q$ to be identical.

In order to express (28), we just have to check for each $\dot{x}_i$ that the ball of radius δ^r around it is a model of $\rho(\dot{x}_i)$. Since the size of this ball is at most δ^r, we can use the INDUCED notation. There is, however, a technical problem: We basically wish to check whether $\mathcal{G}[N_r(a)] \in \{\mathcal{H} \mid \mathcal{H} \models \rho(a)\}$ holds for a given a, but $\{\mathcal{H} \mid \mathcal{H} \models \rho(a)\}$ obviously depends on a—which is not compatible with the INDUCED notation. Fortunately, this problem can be fixed: For $i \in \mathbb{N}$ let $Q_i = \{\mathcal{H} \mid i \leq \|\mathcal{H}\|, a \text{ is the } i\text{th element of } |\mathcal{H}| \text{ with respect to } <^{\mathcal{H}}, \mathcal{H} \models \rho(a)\}$ (recall that for the INDUCED notation the set Q can be arbitrarily

complex). If we know for some element $a \in |\mathcal{G}|$ that it is the ith element in $N_r(a)$, then our problematic test can be replaced by $\mathcal{G}[N_r(a)] \in Q_i$. Since testing whether a is the ith element in $N_r(a)$ is possible using a formula like $\iota_i(a) = \exists^{=i-1} b(b < a \wedge \gamma_{\bar{d}(a,b)\leq r})$, we get the following complete formula $\phi_{\psi,\delta}$:

$$\exists \dot{x}_1 \cdots \exists \dot{x}_k \, \exists \dot{y}_1^1 \cdots \exists \dot{y}_k^{\delta^r} \Big(\bigwedge_{i\neq j} \bigwedge_{p,q\in\{1,\ldots,\delta^r\}} \dot{y}_i^p \neq \dot{y}_j^p \wedge \bigwedge_{i=1}^k \exists x \big(x = \dot{x}_i \wedge \{\dot{y}_i^1, \ldots, \dot{y}_i^{\delta^r}\} = \{y \mid \gamma_{\bar{d}(x,y)\leq r}\} \wedge \bigvee_{i=1}^{\delta^r} (\iota_i(x) \wedge \text{INDUCED}^{\text{size}\leq\delta^r}\{y \mid \gamma_{\bar{d}(x,y)\leq r}\} \in Q_i)\big)\Big).$$

This formula uses only a constant number of strong variables. Its strong quantifier rank would also be constant except that the formula $\gamma_{\bar{d}(x,y)\leq r}$ uses r nested (strong) quantifiers (but only 2 variables). This means that the strong quantifier rank of $\phi_{\psi,\delta}$ will be $O(\text{locality-rank}(\psi))$. □

4.5. Bounded Strong-Rank Description of Embedding Graphs of Constant Tree Width or Constant Tree Depth

For our final example, a graph $H = (V(H), E(H))$ embeds into another graph $G = (V(G), E(G))$ if there is an injective mapping $\iota\colon V(H) \to V(G)$ such that for all $(u, v) \in E(H)$ we have $(\iota(u), \iota(v)) \in E(G)$. We wish to show that the embedding problems for graphs of bounded tree depth or bounded tree width, parameterized by the to-be-embedded graphs, lie in para-AC^0 and para-AC$^{0\uparrow}$, see Theorem 8 below. Let us first briefly review the underlying definitions. A tree decomposition of H is a tree $T = (V(T), E(T))$ (a connected, acyclic, undirected graph) together with a mapping B that assigns a subset of $V(H)$ to each node in $V(T)$. These subsets are called bags and must have two properties: First, for every edge $\{u, v\} \in E(H)$ there must be a node $n \in V(T)$ with $u, v \in B(n)$. Second, for each vertex $v \in V(H)$ the set $\{n \in V(T) \mid v \in B(n)\}$ must be nonempty and connected in T. Let $\text{width}(B) = \max_{n\in V(T)} |B(n)| - 1$. The tree width $\text{tw}(H)$ of H is the minimum width of any tree decomposition for it. We call (T, B) a tree-depth decomposition if T can be rooted in such a way that if u lies on the path from some vertex v to the root, then $B(u) \subsetneq B(v)$. The tree depth $\text{td}(H)$ is the minimum width of a tree-depth decomposition (T, B) of H plus 1. Note that this width is an upper bound on $\text{depth}(T)$, the depth of T, and that this definition is slightly different from the one briefly mentioned in the introduction (but gives the same class).

The problems $\text{p}_H\text{-EMB}_{\text{td}(H)\leq c}$ and $\text{p}_H\text{-EMB}_{\text{tw}(H)\leq c}$, for a fixed number c (not a parameter), contains all pairs $(G, \text{num}(H)) \in \text{STRUC}[(E^2)] \times \mathbb{N}$ such that there exists an embedding of H into G and $\text{td}(H) \leq c$ and $\text{tw}(H) \leq c$, respectively.

Theorem 8 ([6,7]). $\text{p}_H\text{-EMB}_{\text{td}(H)\leq c} \in$ para-AC^0 *and* $\text{p}_H\text{-EMB}_{\text{tw}(H)\leq c} \in$ para-AC$^{0\uparrow}$ *for each c.*

Proof. We present a family $(\phi_{H,T,B})$ of τ-formulas (where $\tau = (E^2, <^2, \text{SUCC}^1, \text{ADD}^3, \text{MULT}^3, 0^0)$ is the arithmetic signature of graphs) indexed by graphs H together with any tree decomposition (T, B) of H (without bounds on the depth or width) that describe the embedding problem. More precisely, we show the following:

Claim 5. *There is a family* $(\phi_{H,T,B})_H \in \text{STRUC}[\tau]$,$(T, B)$ is a tree decomposition of H *such that:*

1. $\mathcal{G} \models \phi_{H,T,B}$ *if and only if* H *embeds into* $\mathcal{G}$ *(more precisely, into* $(|\mathcal{G}|, E^{\mathcal{G}})$*).*
2. $\text{strong-qr}(\phi_{H,T,B}) = \text{depth}(T)$.
3. $|\text{strong-bound}(\phi_{H,T,B})| = \text{width}(B) + 1$.

Proof. Before we present the formula, we define what we will call a consistent numbering of the vertices of H. It is a mapping $p\colon V(H) \to \{1, \ldots, m\}$, where m is the maximum bag size of the decomposition (so $m = \text{width}(B) + 1$). The number $p(v)$ for $v \in V(H)$ can be thought as the "position" or "index" of v in all bags that contain it, that is, we require that for any bag $B(n) = \{b_1, \ldots, b_{|B(n)|}\}$ the values $p(b_1), \ldots, p(b_{|B(n)|})$ are all different.

(Phrased differently, p restricted to any bag is injective.) Such a consistent numbering can be obtained as follows: First, assign the numbers 1 to $|B(r)|$ to the elements of the root bag $B(r)$. Now, consider a child c of the root r in T. The bag $B(c)$ may miss some of the elements of $B(r)$ and there may be some new elements. For each new element e, let $p(e)$ be a different number from the set $\{1, \dots, m\} \setminus \{p(v) \mid v \in B(r) \cap B(c)\}$ and note that we will not run out of numbers. We assign numbers to all elements in the bags of the children of the root in this way and, then, we recursively use the same method for the children's children and so on. Note that, not only, we do not run out of numbers, but the consistency condition is also met: Once an element drops out of a bag, we will never see it again in a later bag and, hence, we cannot inadvertently assign a different number to it later on.

As a running example, we will use the graph H and the tree decomposition (T, B) of it from Figure 1.

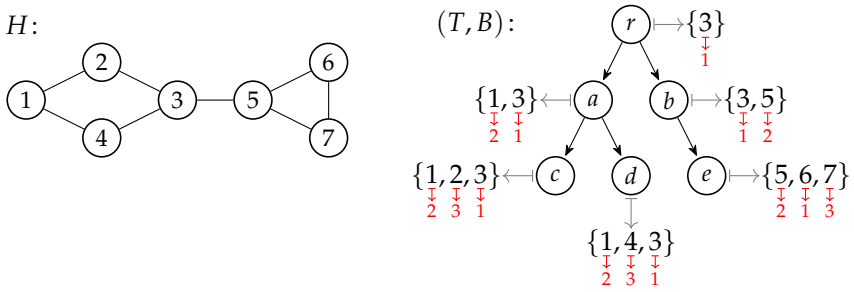

Figure 1. An example graph H together with a tree decomposition for it, consisting of the tree T and the bag function B indicated using the small gray mapping arrows. A consistent numbering p is indicated in red.

Let us now define $\phi_{H,T,B}$. We may assume $V(H) = \{1, \dots, |V(H)|\}$. The first step is to bind all vertices of H to weak variables using $\exists \dot{x}_1 \cdots \exists \dot{x}_{|V(H)|} \bigwedge_{i \neq j} \dot{x}_i \neq \dot{x}_j \wedge \psi$, where ψ must now express that the bound elements form an embedding. To achieve this, we build ψ recursively, starting at the root r of T and $\psi = \psi_r$.

For our example, we would have:

$$\phi_{H,T,B} = \exists \dot{x}_1 \cdots \exists \dot{x}_7 \bigwedge_{i \neq j} \dot{x}_i \neq \dot{x}_j \wedge \psi_r.$$

For any node $n \in V(T)$, let the elements of the set $B(n)$ be named b_1 to b_s (these are just temporary names that have nothing to do with the consisting numbering p) and let the first t of them be new, that is, not present in the parent bag (for the root, $s = t$ and all elements are new; if there are no new elements, $t = 0$). The formula ψ_n will now express the following: First, it binds the new elements using strong variables that are made equal to the weak variables representing the elements in the input structure. This makes the new strong variables disjoint from one another and also from all other (images of) vertices of H. Second, we check that for all edges $\{x, y\} \in E(H)$ between elements x and y of $B(n)$, that their images (which we have bound to strong variables) are also connected in the input structure. Third, we require that these properties also hold for all children of n. In symbols, we set:

$$\begin{aligned} \psi_n = \exists v_{p(b_1)} \cdots \exists v_{p(b_t)} \big(& v_{p(b_1)} = \dot{x}_{b_1} \wedge \cdots \wedge v_{p(b_t)} = \dot{x}_{b_t} \wedge \\ & \textstyle\bigwedge_{x,y \in B(n), \{x,y\} \in E(H)} E v_{p(x)} v_{p(y)} \wedge \\ & \textstyle\bigwedge_{c \in \text{children}(n)} \psi_c \big). \end{aligned}$$

For our example, let us start with the root r. Here, we have $B(r) = \{3\}$ and $p(3) = 1$ and there are no edges between the vertices in the bag (there is just one vertex, after all). This yields: $\psi_n = \exists v_1 (v_1 = \dot{x}_3 \wedge \psi_a \wedge \psi_b)$.

For the node a, a new node (1) enters the bag $B(a)$ with index $2 = p(1)$, but there are no intra-bag edges, so $\psi_a = \exists v_2 (v_2 = \dot{x}_1 \wedge \psi_c \wedge \psi_d)$.

For the node b the situation is very similar, but there is now an edge $\{3,5\}$ in H. This means that we must check that the nodes v_1, representing 3, and v_2, representing 5, are connected in the input structure. This yields $\psi_b = \exists v_2(v_2 = \dot{x}_5 \wedge Ev_1v_2 \wedge \psi_e)$.

For the node c, we only have one new node (2) with a new index ($3 = p(2)$), but now there are two intra-bag edges in H, namely $\{1,2\} \in E(H)$ and $\{2,3\} \in E(H)$. This yields: $\psi_c = \exists v_3(v_3 = \dot{x}_2 \wedge Ev_3v_1 \wedge Ev_3v_2)$, where Ev_1v_3 checks whether for $\{2,3\} \in E(H)$ there is a corresponding edge in the input structure (recall that $p(2) = 3$ and $p(3) = 1$) and Ev_3v_2 checks the same for $\{1,2\}$.

In a similar way, we get $\psi_d = \exists v_3(v_3 = \dot{x}_4 \wedge Ev_3v_1 \wedge Ev_3v_2)$ and observe that the only difference is that v_3 is made equal to $\dot{x}_4$ instead of $\dot{x}_2$, the rest is the same.

Finally, for the node e, we bind two strong variables since there are two new vertices (6 and 7), but we reuse variable v_1 for 6 since the vertex 3 that used to have index 1 has dropped out of the bag. We get

$$\psi_e = \exists v_1 \exists v_3(v_1 = \dot{x}_6 \wedge v_3 = \dot{x}_7 \wedge Ev_1v_2 \wedge Ev_1v_3 \wedge Ev_2v_3).$$

Putting it all together, we have the following $\phi_{H,T,B}$, whose structure closely mirrors T's:

$$\begin{aligned}
&\exists \dot{x}_1 \cdots \exists \dot{x}_7 \textstyle\bigwedge_{i \neq j} \dot{x}_i \neq \dot{x}_j \wedge \\
&\qquad \exists v_1(v_1 = \dot{x}_3 \wedge \\
&\qquad\quad \exists v_2(v_2 = \dot{x}_1 \wedge \\
&\qquad\qquad \exists v_3(v_3 = \dot{x}_2 \wedge Ev_3v_1 \wedge Ev_3v_2) \wedge \\
&\qquad\qquad \exists v_3(v_3 = \dot{x}_4 \wedge Ev_3v_1 \wedge Ev_3v_2)) \wedge \\
&\qquad\quad \exists v_2(v_2 = \dot{x}_5 \wedge Ev_1v_2 \wedge \\
&\qquad\qquad \exists v_1 \exists v_3(v_1 = \dot{x}_6 \wedge v_3 = \dot{x}_7 \wedge Ev_1v_2 \wedge Ev_1v_3 \wedge Ev_2v_3))).
\end{aligned}$$

It remains to argue that $\phi_{H,T,B}$ has the claimed properties. Clearly, by construction, the strong quantifier rank and number of strongly bound variables are as claimed. The semantic correctness also follows easily from the construction: If the input structure is a model of the formula then, clearly, the assignments of the $\dot{x}_i$ to elements of the universe form an embedding since for every edge $\{u,v\} \in E(H)$ somewhere in the formula we test whether $Ev_{p(u)}v_{p(v)}$ holds where $v_{p(u)}$ is equal to $\dot{x}_u$ and $v_{p(v)}$ to $\dot{x}_v$. The other way round, given a model of the formula, any assignment to the $\dot{x}_i$ that makes it true is an embedding since, first, we require that all $\dot{x}_i$ are different and we require $Ev_{p(u)}v_{p(v)}$ for all $\{u,v\} \in E(H)$. This concludes the proof of the claim. □

With the claim established, we can now easily derive the statement of the theorem. To show $\mathrm{p}_H\text{-}\mathrm{EMB}_{\mathrm{td}(H)\leq c} \in \text{para-AC}^0$, we must present a family $(\phi_H)_{H \in \mathrm{STRUC}[\tau], \mathrm{td}(H) \leq c}$ that describes $\mathrm{p}_H\text{-}\mathrm{EMB}_{\mathrm{td}(H)\leq c}$ and that has bounded quantifier rank. Clearly, we can just set ϕ_H to $\phi_{H,T,B}$ where (T,B) is a tree-depth decomposition of H of depth c (which must exist by the assumption that $\mathrm{td}(H) \leq c$). The second item of the claim immediately tells us that all ϕ_H will have a strong quantifier rank of at most c; and we can use the characterization of para-AC^0 from Fact 1. For the second statement, $\mathrm{p}_H\text{-}\mathrm{EMB}_{\mathrm{tw}(H)\leq c} \in \text{para-AC}^{0\uparrow}$, we use a different family $(\psi_H)_{H \in \mathrm{STRUC}[\tau], \mathrm{tw}(H) \leq c}$, this time setting ψ_H to $\phi_{H,T,B}$ where (T,B) is a tree decomposition of H of width c. Now the third item of the claim gives us the bound on the number of strong variables; and we can use the characterization of para-AC$^{0\uparrow}$ from Theorem 1. □

5. Conclusions

In the present paper, we showed how the color coding technique can be turned into a powerful tool for parameterized descriptive complexity theory. This tool allows us to show that important results from parameterized complexity theory—like the fact that the embedding problem for graphs of bounded tree width lies in FPT—follow just from

the syntactic structure of the formulas that describe the problem. The core contribution of the paper does not lie in proving new membership results for para-AC^0 or para-$AC^{0\uparrow}$, but in explaining why problems lie in these classes in terms of the syntax of formulas describing the problems. Apart from unifying previous results, we also get much shorter and simpler proofs.

In all our syntactic characterizations it was important that variables or color predicates were not allowed to be within a universal scope. The reason was that literals, disjunctions, conjunctions, and existential quantifiers all have what we called the small witness property, which universal quantifiers do not have. However, there are other quantifiers, from more powerful logics that we did not explore, that also have the small witness property. An example are operators that test whether there is a path of length at most k from one vertex to another for some fixed k: if such a path exists, its vertices form a "small witness." Weak variables may be used inside these operators, leading to broader classes of problems that can be described by families of bounded strong quantifier rank. On the other hand, we cannot add the full transitive closure operator TC (for which it is well-known that $\mathrm{FO}[\mathrm{TC}] = \mathrm{NL}$) and hope that Theorems 2 and 3 still hold: If this were the case, we should be able to turn a formula that uses two colors C_1 and C_2 to express that there are two vertex-disjoint paths between two vertices into a $\mathrm{FO}[\mathrm{TC}]$ formula—thus proving the unlikely result that the NP-hard disjoint path problem is in NL.

Another line of inquiry into the descriptive complexity of parameterized problems was already started in the repeatedly cited paper by Chen et al. [4]: They give first syntactic properties for families of formulas describing weighted model checking problems that imply membership in para-AC^0. We believe that it might be possible to base an alternative notion of weak quantifiers on these syntactic properties. Ideally, we would like to prove a theorem similar to Theorem 4 in which there are just more quantifiers that count as weak and, hence, even more families have bounded strong quantifier rank. This would allow us to prove for even more problems that they lie in FPT just because of the syntactic structure of the natural formula families that describe them.

Author Contributions: Formal analysis, M.B. and T.T.; Writing—original draft, T.T.; Writing—review & editing, M.B. The authors have contributed equally. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Fagin, R. Generalized First-Order Spectra and Polynomial-Time Recognizable Sets. *Complex. Comput.* **1974**, *7*, 43–74.
2. Flum, J.; Grohe, M. *Parameterized Complexity Theory*; Texts in Theoretical Computer Science; Springer: Abingdon, UK, 2006; [CrossRef]
3. Alon, N.; Yuster, R.; Zwick, U. Color-Coding. *J. ACM* **1995**, *42*, 844–856. [CrossRef]
4. Chen, Y.; Flum, J.; Huang, X. Slicewise Definability in First-Order Logic with Bounded Quantifier Rank. *arXiv* **2017**, arXiv:1704.03167.
5. Bannach, M.; Tantau, T. Parallel Multivariate Meta-Theorems. In Proceedings of the Eleventh International Symposium on Parameterized and Exact Computation (IPEC 2016), Aarhus, Denmark, 24–26 August 2016; pp. 4:1–4:17. [CrossRef]
6. Chen, H.; Müller, M. The Fine Classification of Conjunctive Queries and Parameterized Logarithmic Space. *ACM Trans. Comput. Theory* **2015**, *7*, 7:1–7:27. [CrossRef]
7. Bannach, M.; Stockhusen, C.; Tantau, T. Fast Parallel Fixed-parameter Algorithms via Color Coding. In Proceedings of the Tenth International Symposium on Parameterized and Exact Computation (IPEC 2015), Patras, Greece, 16–18 September 2015; pp. 224–235. [CrossRef]

8. Das, B.; Enduri, M.K.; Reddy, I.V. On the Parallel Parameterized Complexity of the Graph Isomorphism Problem. In Proceedings of the Twelfth International Conference and Workshop on Algorithms and Computation (WALCOM 2018), Dhaka, Bangladesh, 3–5 March 2018; pp. 252–264. [CrossRef]
9. Pilipczuk, M.; Siebertz, S.; Toruńczyk, S. Parameterized circuit complexity of model-checking on sparse structures. In Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2018), Oxford, UK, 9–12 July 2018; pp. 789–798. [CrossRef]
10. Flum, J.; Grohe, M. Describing Parameterized Complexity Classes. *Inf. Comput.* **2003**, *187*, 291–319. [CrossRef]
11. Chen, Y.; Flum, J. Tree-depth, quantifier elimination, and quantifier rank. In Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2018), Oxford, UK, 9–12 July 2018; pp. 225–234. [CrossRef]
12. Cai, L.; Chen, J. On fixed-parameter tractability and approximability of NP optimization problems. *J. Comput. Syst. Sci.* **1997**, *54*, 465–474. [CrossRef]
13. Papadimitriou, C.; Yannakakis, M. Optimization, approximation, and complexity classes. *J. Comput. Syst. Sci.* **1991**, *43*. [CrossRef]
14. Kolaitis, P.; Thakur, M. Logical definability of NP optimization problems. *Inf. Comput.* **1994**, *115*, 321–353. [CrossRef]
15. Vollmer, H. *Introduction to Circuit Complexity—A Uniform Approach*; Texts in Theoretical Computer Science, An EATCS Series; Springer: Berlin/Heidelberg, Germany, 1999. [CrossRef]
16. Immerman, N. DSPACE$[n^k]$ = VAR$[k+1]$. In Proceedings of the Sixth Annual Structure in Complexity Theory Conference, Chicago, IL, USA, 30 June–3 July 1991; pp. 334–340. [CrossRef]
17. Immerman, N. *Descriptive Complexity*; Springer: Berlin/Heidelberg, Germany, 1998. [CrossRef]
18. Hüffner, F.; Wernicke, S.; Zichner, T. Algorithm Engineering for Color-Coding with Applications to Signaling Pathway Detection. *Algorithmica* **2008**, *52*, 114–132. [CrossRef]
19. Bannach, M.; Tantau, T. Computing Hitting Set Kernels By AC^0-Circuits. In Proceedings of the 35th Symposium on Theoretical Aspects of Computer Science (STACS 2018), Caen, France, 28 February–3 March 2018; pp. 9:1–9:14, [CrossRef]
20. Erdős, P.; Rado, R. Intersection theorems for systems of sets. *J. Lond. Math. Soc.* **1960**, *1*, 85–90. [CrossRef]
21. Gaifman, H. On Local and Non-Local Properties. In *Studies in Logic and the Foundations of Mathematics*; Elsevier: Amsterdam, The Netherlands, 1982; pp. 105–135. [CrossRef]