BIROn - Birkbeck Institutional Research Online

Chen, Hubie and Muller, M. (2015) The fine classification of conjunctive queries and parameterized logarithmic space. In: UNSPECIFIED (ed.) PODS '13 Proceedings of the 32nd ACM SIGMOD-SIGACT-SIGAI symposium on Principles of database systems. New York, U.S.: ACM Publications, pp. 309-320. ISBN 9781450320665.

The Fine Classification of Conjunctive Queries and Parameterized Logarithmic Space Complexity

Hubie Chen[*]

Departamento LSI, Facultad de Informática
Universidad del País Vasco, San Sebastián, Spain
and
IKERBASQUE, Basque Foundation for Science, Bilbao, Spain

Moritz Müller[†]

Kurt Gödel Research Center, Universität Wien, Vienna, Austria

Abstract

We perform a fundamental investigation of the complexity of conjunctive query evaluation from the perspective of parameterized complexity. We classify sets of boolean conjunctive queries according to the complexity of this problem. Previous work showed that a set of conjunctive queries is fixed-parameter tractable precisely when the set is equivalent to a set of queries having bounded treewidth. We present a fine classification of query sets up to parameterized logarithmic space reduction. We show that, in the bounded treewidth regime, there are three complexity degrees and that the properties that determine the degree of a query set are bounded pathwidth and bounded tree depth. We also engage in a study of the two higher degrees via logarithmic space machine characterizations and complete problems. Our work yields a significantly richer perspective on the complexity of conjunctive queries and, at the same time, suggests new avenues of research in parameterized complexity.

1 Introduction

Conjunctive queries are the most basic and most heavily studied database queries, and can be formalized logically as formulas consisting of a sequence of existentially quantified variables, followed by a conjunction of atomic formulas. Ever since the landmark

---

[*]E-mail address: `hubie.chen@ehu.es`

[†]E-mail address: `moritz.mueller@univie.ac.at`

1

1977 article of Chandra and Merlin [4], complexity-theoretic aspects of conjunctive queries have been a research subject of persistent and enduring interest which continues to the present day (as a sampling, we point to the works [1, 22, 27, 18, 19, 20, 9, 28, 25]; see the discussions and references therein for more information). The problem of evaluating a conjunctive query on a relational database is equivalent to a number of well-known problems, including conjunctive query containment, the homomorphism problem on relational structures, and the constraint satisfaction problem [4, 22]. That this evaluation problem appears in many equivalent guises attests to the fundamental and primal nature of this problem, and it has correspondingly been approached and studied from a wide variety of perspectives and motivations. The resulting literature has not only been fruitful in terms of continually providing insights into and notions for understanding conjunctive queries themselves, but has also meaningfully fed back into a richer understanding of computational complexity theory at large, and of common complexity classes in particular. This is witnessed by the observation that various flavors of conjunctive query evaluation are used as prototypical complete problems for complexity classes such as NP and W[1] (refer, for example, to the books by Creignou, Khanna, and Sudan [8] and by Flum and Grohe [17], respectively). Another example of this phenomenon is the work showing LOGCFL-completeness of evaluating acyclic conjunctive queries (as well as of many related problems) due to Gottlob, Leone, and Scarcello [18].

As has been eloquently articulated in the literature [27], the employment of classical complexity notions such as polynomial-time tractability to grade the complexity of conjunctive query evaluation is not totally satisfactory. For in the context of databases, the typical scenario is the evaluation of a relatively short query on a relatively large database; this suggests a notion of time complexity wherein a non-polynomial dependence on the query may be tolerated, so long as the dependence on the database is polynomial. Computational complexity theory has developed and studied precisely such a relaxation of polynomial-time tractability, called *fixed-parameter tractability*, in which arbitrary dependence in a *parameter* is permitted; in our query evaluation setting, the query size is normally taken as the parameter. The class of such tractable problems is denoted by FPT. Fixed-parameter tractability is the base tractability notion of *parameterized complexity theory*, a comprehensive theory for studying problems where each instance has an associated parameter. As a parameterized problem, conjunctive query evaluation is complete for the parameterized complexity class W[1] [27, 17]; the property of W[1]-hardness plays, in the parameterized setting, a role similar to that played by NP-hardness in the classical setting.

Due to the general intractability of conjunctive query evaluation, a recurring theme in the study of conjunctive queries is the identification of structural properties that provide tractability; such properties include *acyclicity* and *bounded treewidth* [18, 22]. A natural research issue is to obtain a systematic understanding of what properties ensure tractability, by classifying all sets of queries according to the complexity of the evaluation problem. We focus on boolean conjunctive queries, which, in logical parlance, are queries without free variables. Formally, let Φ be a set of boolean conjunctive queries, and define EVAL(Φ) to be the problem of deciding, given a query $\phi \in \Phi$ and a relational structure $\mathbf{B}$, whether or not ϕ evaluates to true on $\mathbf{B}$. One can then inquire for which sets Φ the problem EVAL(Φ) is tractable. For mathematical convenience, we

use an equivalent formulation of this problem. It is known that each boolean conjunctive query ϕ can be bijectively represented as a relational structure $\mathbf{A}$ in such a way that, for any relational structure $\mathbf{B}$, it holds that ϕ is true on $\mathbf{B}$ if and only if there exists a homomorphism from $\mathbf{A}$ to $\mathbf{B}$ [4]. Hence, the following family of problems is equivalent to the family of problems EVAL(Φ). Let $\mathcal{A}$ be a set of structures, and denote by HOM($\mathcal{A}$) the problem of deciding, given a structure $\mathbf{A} \in \mathcal{A}$ and a second structure $\mathbf{B}$, whether or not there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$. Use p-HOM($\mathcal{A}$) to denote the parameterized version of this problem, where the size of $\mathbf{A}$ is taken as the parameter.

Under the assumption that the structures in $\mathcal{A}$ have bounded arity, Grohe [20] presented a classification of the tractable problems of this form: if the *cores* of $\mathcal{A}$ have bounded treewidth, then the problem p-HOM($\mathcal{A}$) is fixed-parameter tractable; otherwise, the problem p-HOM($\mathcal{A}$) is W[1]-hard. The *core* of a structure can be intuitively thought of as a smallest equivalent structure. Grohe's classification thus shows that, in the studied setting, the condition of bounded treewidth is the *only* property guaranteeing tractability (assuming FPT $\neq$ W[1]). Recall that treewidth is a graph measure which, intuitively speaking, measures the similitude of a graph to a tree, with a lower measure indicating a higher degree of similarity. The assumption of bounded arity provides robustness in that translating between two reasonable representations of structures can be done efficiently; this is in contrast to the case of unbounded arity, where the choice of representation can dramatically affect complexity [5].

The present article was motivated by the following fundamental research question: *What algorithmic/complexity behaviors of conjunctive queries are possible, within the regime of fixed-parameter tractability?* That is, we endeavored to obtain a finer perspective on the parameterized complexity of conjunctive queries, and in particular, on the possible sources of tractability thereof, by presenting a classification result akin to Grohe's, but for queries that are fixed-parameter tractable. As is usual in computational complexity, we make use of a weak notion of reduction in order to be able to make fine distinctions within the tractable zone. Logarithmic space computation is a common machine-based mode of computation that is often used to make distinctions within polynomial time; correspondingly, we adopt *parameterized logarithmic space computation*, which is obtained by relaxing logarithmic space computation much in the way that fixed-parameter tractability is obtained by relaxing polynomial time, as the base complexity class and as the reduction notion used in our investigation.

We present a classification theorem that comprehensively describes, for each set $\mathcal{A}$ of structures having bounded arity and bounded treewidth, the complexity of the problem p-HOM($\mathcal{A}$), up to parameterized logarithmic space reducibility (Section 3). Let $\mathcal{T}$ denote the set of all graphs that are trees, $\mathcal{P}$ denote the set of all graphs that are paths, and, for a set of structures $\mathcal{A}$, let $\mathcal{A}^*$ denote the set of structures obtainable by taking a structure $\mathbf{A} \in \mathcal{A}$ and adding each element of $\mathbf{A}$ as a relation. Our theorem shows that precisely three degrees of behavior are possible: such a problem p-HOM($\mathcal{A}$) is either equivalent to p-HOM($\mathcal{T}^*$), equivalent to p-HOM($\mathcal{P}^*$), or is solvable in parameterized logarithmic space (Theorem 3.1). Essentially speaking, bounded pathwidth and bounded tree depth are the properties that determine which of the three cases hold; as with treewidth, both pathwidth and tree depth are graph measures that associate a natural number with each graph. A key component of our classification theorem's proof is a reduction that, in effect, allows us to prove hardness results on a

problem p-HOM($\mathcal{A}$) based on the hardness of p-HOM($\mathcal{M}^*$) where $\mathcal{M}$ consists of certain graph minors derived from $\mathcal{A}$ (Lemma 3.6). The proof of our classification theorem utilizes this reduction in conjunction with excluded minor characterizations of graphs of bounded pathwidth and of bounded tree depth. We remark that, in combination with the *excluded grid theorem* from graph minor theory, the discussed reduction can be employed to readily derive Grohe's classification from the hardness of the *colored grid homomorphism problem*; this hardness result was presented by Grohe, Schwentick, and Segoufin [21]. A fascinating aspect of our classification theorem, which is shared with that of Grohe, is that natural graph-theoretic conditions–in our case, those of bounded pathwidth and bounded tree depth–arise naturally as the relevant properties that are needed to present our classification. This theorem also widens the interface among conjunctive queries, graph minor theory, and parameterized complexity that is present in the discussed work [21, 20].

Given that the problems p-HOM($\mathcal{P}^*$) and p-HOM($\mathcal{T}^*$) are the *only* problems (up to equivalence) above parameterized logarithmic space that emerge from our classification, we then seek a richer understanding of these problems. In particular, we engage in a study of the complexity classes that these problems define: we study the class of problems that reduce to p-HOM($\mathcal{P}^*$), and likewise for p-HOM($\mathcal{T}^*$) (Sections 4 and 5). Following a time-honored tradition in complexity theory, we present machine-based definitions of these classes, which classes we call PATH and TREE, respectively. The machine definition of PATH comes from recent work of Elberfeld, Stockhusen, and Tantau [12] and is based on nondeterministic Turing machines satisfying two simultaneous restrictions: first, that only parameterized logarithmic space is consumed; second, that the number of nondeterministic bits used is bounded, namely, by the product of the logarithm of the input size and a constant depending on the parameter. The machine characterization of TREE is similar, but it is based on alternating Turing machines where, in addition to the nondeterministic bits permitted previously, a parameter-dependent number of conondeterministic bits may also be used. In addition to proving that the problems p-HOM($\mathcal{P}^*$) and p-HOM($\mathcal{T}^*$) are complete for the machine-defined classes, we also prove that for any set of structures $\mathcal{A}$ having bounded pathwidth, the parameterized *embedding* problem p-EMB($\mathcal{A}$) is in PATH, and prove an analogous result for structures of bounded treewidth and the class TREE.

In the final section of the paper, we present a fine classification for the problem of counting homomorphisms which is analogous to our classification for the homomorphism problem (Section 6).

Our work shows that the complexity classes PATH and TREE are heavily populated with complete problems, and, along with the recent work [12], suggests the further development of the study of space-bounded parameterized complexity [15, 6] and, speaking more broadly, the study of complexity classes within FPT, which may include classes based on circuit or parallel models of computation. We can mention the following natural structural questions. Are either of the classes PATH or TREE closed under complement? Can any evidence be given either in favor of or against such closure? Even if the classes PATH and TREE are not closed under complement, could it be that co-PATH $\subseteq$ TREE? Another avenue for future research is to develop the theory of the degrees of counting problems identified by our counting classification. We shall mention some further open questions in the final section.

2 Preliminaries

For $n \in \mathbb{N}$ we define $[n] := \{1, \ldots, n\}$ if $n > 0$ and $[0] := \emptyset$. We write $\{0,1\}^{\leq n}$ for the set of binary strings $x \in \{0,1\}^*$ of length $|x| \leq n$; we have $\{0,1\}^{\leq 0} = \{\lambda\}$ where λ is the empty string.

2.1 Structures, homomorphisms and cores

Structures

A *vocabulary* τ is a finite set of relation symbols, where each $R \in \tau$ has an associated *arity* $\mathsf{ar}(R) \in \mathbb{N}$. A τ-*structure* $\mathbf{A}$ consists of a nonempty finite set A, its *universe*, together with an *interpretation* $R^{\mathbf{A}} \subseteq A^{\mathsf{ar}(R)}$ of every $R \in \tau$. Let us emphasize that, in this article, we consider only finite structures. A *substructure (weak substructure)* of $\mathbf{A}$ is a structure *induced* by a nonempty subset X of A, i.e. the structure $\langle X \rangle^{\mathbf{A}}$ with universe X that interprets every $R \in \tau$ by (respectively, a subset of) $X^{\mathsf{ar}(R)} \cap R^{\mathbf{A}}$. A *restriction* of a structure is obtained by forgetting the interpretations of some symbols, and an *expansion* of a structure is obtained by adding interpretations of some symbols. We view *directed graphs* as $\{E\}$-structures $\mathbf{G} := (G, E^{\mathbf{G}})$ for binary E; $\mathbf{G}$ is a *graph* if $E^{\mathbf{G}}$ is irreflexive and symmetric. Note that a weak substructure of a graph is a subgraph. The graph *underlying* a directed graph $\mathbf{G}$ without loops (i.e. with irreflexive $E^{\mathbf{G}}$) is obtained by replacing $E^{\mathbf{G}}$ with its symmetric closure. We shall be concerned with the following classes of structures.

- For $k \geq 2$, the structure $\overrightarrow{\mathbf{P}_k}$ has universe $[k]$ and edge relation $\{(i, i+1) \mid i \in [k-1]\}$. The class $\overrightarrow{\mathcal{P}}$ of *directed paths* consists of the structures that are isomorphic to a structure of this form.

  Let $\mathbf{P}_k$ be the graph underlying $\overrightarrow{\mathbf{P}_k}$. The class $\mathcal{P}$ of *paths* consists of the structures that are isomorphic to a structure of this form.

- For $k \geq 2$, the structure $\overrightarrow{\mathbf{C}_k}$ has universe $[k]$ and edge relation $\{(i, i+1) \mid i \in [k-1]\} \cup \{(k,1)\}$. The class $\overrightarrow{\mathcal{C}}$ of *directed cycles* consists of the structures that are isomorphic to a structure of this form.

  Let $\mathbf{C}_k$ be the graph underlying $\overrightarrow{\mathbf{C}_k}$. The class $\mathcal{C}$ of *cycles* consists of the structures that are isomorphic to a structure of this form.

- For $k \geq 0$, the structure $\overrightarrow{\mathbf{B}_k}$ has universe $\{0,1\}^{\leq k}$ and binary relations $S_i^{\overrightarrow{\mathbf{B}_k}} = \{(x, xi) \mid x \in \{0,1\}^{\leq k-1}\}$ for $i \in \{0,1\}$. The class $\overrightarrow{\mathcal{B}}$ consists of the structures that are isomorphic to a structure of this form.

  Let $\mathbf{T}_k$ be the graph underlying the directed graph $(\{0,1\}^{\leq k}, S_0^{\overrightarrow{\mathbf{B}_k}} \cup S_1^{\overrightarrow{\mathbf{B}_k}})$.

  Let $\mathbf{B}_k$ be the structure with universe $\{0,1\}^{\leq k}$ and binary relations $S_0^{\mathbf{B}_k}$, $S_1^{\mathbf{B}_k}$ defined to be the symmetric closures of the relations $S_0^{\overrightarrow{\mathbf{B}_k}}$, $S_1^{\overrightarrow{\mathbf{B}_k}}$, respectively. The class $\mathcal{B}$ consists of the structures that are isomorphic to a structure of this form.

- Finally, $\mathcal{T}$ is the class of *trees*, that is, the class of connected, acyclic graphs.

5

A class of structures $\mathcal{A}$ has *bounded arity* if there exists a $r \in \mathbb{N}$ such that any relation symbol interpreted in any structure $\mathbf{A} \in \mathcal{A}$ has arity at most r.

Homomorphisms

Let $\mathbf{A}, \mathbf{B}$ be structures. A *homomorphism* from $\mathbf{A}$ to $\mathbf{B}$ is a function $h : A \to B$ such that for all $R \in \tau$ and for all $\bar{a} = (a_1, \ldots, a_{\mathsf{ar}(R)}) \in R^{\mathbf{A}}$ it holds that $h(\bar{a}) \in R^{\mathbf{B}}$ where we write $h(\bar{a}) = (h(a_1), \ldots, h(a_{\mathsf{ar}(R)}))$. A *partial* homomorphism from $\mathbf{A}$ to $\mathbf{B}$ is the empty set or a homomorphism from a substructure of $\mathbf{A}$ to $\mathbf{B}$; equivalently, this is a partial function h from A to B that is a homomorphism from $\langle \mathsf{dom}(h) \rangle^{\mathbf{A}}$ to $\mathbf{B}$ if the domain $\mathsf{dom}(h)$ of h is not empty. As has become usual in our context, by an *embedding* we mean an injective homomorphism.

A structure $\mathbf{A}$ is a *core* if all homomorphisms from $\mathbf{A}$ to $\mathbf{A}$ are embeddings. Every structure $\mathbf{A}$ maps homomorphically to a weak substructure of itself which is a core. This weak substructure is unique up to isomorphism and called the core *of* $\mathbf{A}$ (cf. [13]). For a set of structures $\mathcal{A}$ we let $\mathsf{core}(\mathcal{A})$ denote the set of cores of structures in $\mathcal{A}$. It is not hard to see that two structures $\mathbf{A}, \mathbf{B}$ are homomorphically equivalent (that is, there are homomorphisms in both directions) if and only if they have the same core.

When $\mathbf{A}$ is a structure, we use $\mathbf{A}^*$ to denote its expansion that interprets for every $a \in A$ a fresh unary relation symbol C_a by $C_a^{\mathbf{A}^*} = \{a\}$. For a class of structures $\mathcal{A}$ we let

$$\mathcal{A}^* := \{\mathbf{A}^* \mid \mathbf{A} \in \mathcal{A}\}.$$

Example 2.1. The following facts are straightforward to verify. Trees with at least two vertices and cycles of even length have a single edge as core, and so do cycles of even length. Cycles of odd length are cores, and so are directed paths. Structures of the form $\mathbf{A}^*$ are cores.

2.2 Notions of width

We rely on Bodlaender's survey [3] as a general reference for the notions of treewidth and pathwidth. Tree depth was introduced in [26].

A *tree-decomposition* of a graph $\mathbf{G} = (G, E^{\mathbf{G}})$ is a pair of a tree $\mathbf{T}$ and a family of *bags* $X_t \subseteq G$ for $t \in T$ such that $G = \bigcup_{t \in T} X_t$, $E^{\mathbf{G}} \subseteq \bigcup_{t \in T} X_t^2$ and $X_t \cap X_{t'} \subseteq X_{t''}$ whenever t'' lies on the simple path from t to t'; it is called a *path-decomposition* if $\mathbf{T}$ is a path; its *width* is $\max_{t \in T} |X_t| - 1$.

The *treewidth* $\mathsf{tw}(\mathbf{G})$ of $\mathbf{G}$ is the minimum width of a tree-decomposition of $\mathbf{G}$. The *pathwidth* $\mathsf{pw}(\mathbf{G})$ of $\mathbf{G}$ is the minimum width of a path-decomposition of $\mathbf{G}$.

By a *rooted tree* $\mathbf{T}$ we mean an expansion $(T, E^{\mathbf{T}}, root^{\mathbf{T}})$ of a tree $(T, E^{\mathbf{T}})$ by a unary relation symbol *root* interpreted by a singleton containing the *root*. The *tree depth* $\mathsf{td}(\mathbf{G})$ of $\mathbf{G}$ is the minimum $h \in \mathbb{N}$ such that every connected component of $\mathbf{G}$ is a subgraph of the closure of some rooted tree of height h. Here, the *closure* of a rooted tree is obtained by adding an edge from t to t' whenever t lies on the simple path from the root to t'.

The tree depth $\mathsf{td}(\mathbf{A})$ of an arbitrary structure $\mathbf{A}$ is the tree depth of its *Gaifman graph*: it has vertices A and an edge between a and a' if and only if a and a' are

different and occur together in some tuple in some relation in $\mathbf{A}$. The notions $\mathsf{pw}(\mathbf{A})$ and $\mathsf{tw}(\mathbf{A})$ are similarly defined.

A class $\mathcal{A}$ of structures has *bounded tree depth* if there is $w \in \mathbb{N}$ such that $\mathsf{td}(\mathbf{A}) \leq w$ for all $\mathbf{A} \in \mathcal{A}$. Having bounded pathwidth or treewidth is similarly explained. It is not hard to see that bounded pathwidth is implied by bounded tree depth, and, trivially, bounded treewidth is implied by bounded pathwidth. The converse statements fail:

Example 2.2. The class $\mathcal{P}$ has unbounded tree depth and bounded pathwidth (cf. [26, Lemma 2.2]). The class $\mathcal{B}$ has unbounded pathwidth and bounded treewidth (see. e.g. [3, Theorem 67]).

Such classes are characterized as those excluding certain minors as follows. The first two statements are well-known from Robertson and Seymour's graph minor series (cf. [3, Theorems 12,13]) and the third is from [2, Theorem 4.8].

Theorem 2.3. *Let $\mathcal{C}$ be a class of graphs.*

1. *(Excluded Grid Theorem) $\mathcal{C}$ has bounded treewidth if and only if $\mathcal{C}$ excludes some grid as a minor.*

2. *(Excluded Tree Theorem) $\mathcal{C}$ has bounded pathwidth if and only if $\mathcal{C}$ excludes some tree as a minor.*

3. *(Excluded Path Theorem) $\mathcal{C}$ has bounded tree depth if and only if $\mathcal{C}$ excludes some path as a minor.*

A class of graphs $\mathcal{C}$ *excludes* a graph $\mathbf{M}$ *as a minor* if $\mathbf{M}$ is not a minor of any graph in $\mathcal{C}$. Recall, $\mathbf{M}$ is a *minor* of a graph $\mathbf{G}$ if there exists a *minor map* μ from $\mathbf{M}$ to $\mathbf{G}$, that is, a family $(\mu(m))_{m \in M}$ of pairwise disjoint, non-empty, connected subsets of G such that for all $(m, m') \in E^{\mathbf{M}}$ there are $v \in \mu(m)$ and $v' \in \mu(m')$ with $(v, v') \in E^{\mathbf{G}}$.

It is easy to verify that $\mathsf{td}, \mathsf{pw}, \mathsf{tw}$ are monotone with respect to the minor pre-order, that is, e.g. $\mathsf{td}(\mathbf{G}) \geq \mathsf{td}(\mathbf{M})$ for every minor $\mathbf{M}$ of $\mathbf{G}$. Example 2.2 thus gives the (easy) directions from left to right in the above theorem.

2.3 Parameterized complexity

Turing machines

We identify (classical) problems with sets $Q \subseteq \{0, 1\}^*$ of finite binary strings. We use Turing machines with a (read-only) input tape and several worktapes as our basic model of computation. We will consider nondeterministic and alternating Turing machines with binary nondeterminism and co-nondeterminism. For concreteness, let us agree that a nondeterministic machine has a special *(existential) guess state*; a configuration with the guess state has two successor configurations obtained by changing the guess state to one out of two further distinguished states s_0, s_1. An alternating machine may additionally have a *universal guess state* that follows a similar convention. For a function $f : \{0, 1\}^* \to \mathbb{N}$ we say that $\mathbb{A}$ *uses f (co-)nondeterministic bits* if for every input $x \in \{0, 1\}^*$ every run of $\mathbb{A}$ on x contains at most $f(x)$ many configurations with the existential (respectively, universal) guess state.

Fixed-parameter (in)tractability

A *parameterized problem* (Q, κ) is a pair of a classical problem $Q \subseteq \{0,1\}^*$ and a logarithmic space computable *parameterization* $\kappa : \{0,1\}^* \to \mathbb{N}$ associating with any instance $x \in \{0,1\}^*$ its *parameter* $\kappa(x) \in \mathbb{N}$.[1] A Turing machine is *fpt-time bounded (with respect to κ)* if on input $x \in \{0,1\}^*$ it runs in time $f(\kappa(x)) \cdot |x|^{O(1)}$ where $f : \mathbb{N} \to \mathbb{N}$ is a computable function. The class FPT (para-NP) contains the parameterized problems (Q, κ) such that Q is decided (accepted) by an fpt-time bounded deterministic (nondeterministic) Turing machine. An *fpt-reduction* from (Q, κ) to (Q', κ') is a reduction $R : \{0,1\}^* \to \{0,1\}^*$ from Q to Q' that is computable by a fpt-time bounded (with respect to κ) Turing machine and such that $\kappa' \circ R \leq f \circ \kappa$ for some computable f.

We are concerned with homomorphism and embedding problems associated with classes of structures $\mathcal{A}$.

p-Hom($\mathcal{A}$)	
Instance:	A pair of structures $(\mathbf{A}, \mathbf{B})$ where $\mathbf{A} \in \mathcal{A}$.
Parameter:	$|\mathbf{A}|$.
Problem:	Is there a homomorphism from $\mathbf{A}$ into $\mathbf{B}$?

p-Emb($\mathcal{A}$)	
Instance:	A pair of structures $(\mathbf{A}, \mathbf{B})$ where $\mathbf{A} \in \mathcal{A}$.
Parameter:	$|\mathbf{A}|$.
Problem:	Is there an embedding from $\mathbf{A}$ into $\mathbf{B}$?

These problem definitions exemplify how we present parameterized problems. More formally, the parameterization indicated is the function that maps a string encoding a pair of structures $(\mathbf{A}, \mathbf{B})$ to $|\mathbf{A}|$, and any other string to, say, 0. Here, $|\mathbf{A}| := |\tau| + |A| + \sum_{R \in \tau} |R^A| \cdot \mathrm{ar}(R)$ is the *size* of $\mathbf{A}$; note that the length of a reasonable binary encoding of $\mathbf{A}$ is $O(|\mathbf{A}| \cdot \log |A|)$ (cf. [14]).

The theory of parameterized intractability is centered around the W-hierarchy, which consists of the classes $\mathrm{W}[1] \subseteq \mathrm{W}[2] \subseteq \cdots \subseteq \mathrm{W}[\mathrm{P}]$. The class $\mathrm{W}[\mathrm{P}]$ contains the parameterized problems (Q, κ) that are accepted by nondeterministic Turing machines that are fpt-time bounded with respect to κ and use $f(\kappa(x)) \cdot \log |x|$ many nondeterministic bits. We refer to the monographs [17, 11] for more information about the W-hierarchy. It is well-known that, when $\mathcal{A}$ is a decidable class of structures, the problems p-Hom($\mathcal{A}$) and p-Emb($\mathcal{A}$) are contained in $\mathrm{W}[1]$; when $\mathcal{A}$ is the e.g. class of cliques, these problems are $\mathrm{W}[1]$-hard and hence $\mathrm{W}[1]$-complete under fpt-reductions.

Parameterized logarithmic space

A Turing machine is *parameterized logarithmic space bounded (with respect to κ)*, in short, *pl-space bounded (with respect to κ)* if on input $x \in \{0,1\}^*$ it runs in space $O(f(\kappa(x)) + \log n)$, where $f : \mathbb{N} \to \mathbb{N}$ is some computable function. The class para-L (para-NL) contains the parameterized problems (Q, κ) such that Q is decided

---

[1] Usually polynomial time is allowed to compute κ but as we are interested in parameterized logarithmic space we adopt a more restrictive notion as [12]. Natural parameterizations are often simply projections.

(accepted) by a (non)deterministic Turing machine that is pl-space bounded with respect to κ. Obviously,

$$\text{para-L} \subseteq \text{para-NL} \subseteq \text{FPT} \subseteq \text{W[P]} \subseteq \text{para-NP}.$$

Remark 2.4. Allowing in the above definition space $f(\kappa(x)) \cdot \log|x|$ gives strictly larger classes known as (the stronlgy uniform versions of) XL and XNL. These classes are likely to be incomparable with FPT: they do not contain FPT unless $P = NL$ and contain problems that are even AW[SAT]-hard under fpt-reductions. We shall not be concerned with these classes here and refer the interested reader to [6, 12] for proofs of the mentioned facts and further information. [15] gives some general account of the para- and X-operators.

Let κ be a parameterization. A function $F : \{0,1\}^* \to \{0,1\}^*$ is *implicitly pl-computable (with respect to κ)* if the parameterized problem

BITGRAPH(F)	
Instance:	A triple (x, i, b) where $x \in \{0,1\}^*$, $i \geq 1$, and $b \in \{0,1\}$.
Parameter:	$\kappa(x)$.
Problem:	Does $F(x)$ have length $|F(x)| \geq i$ and ith bit equal to b?

is in para-L. The following is straightforwardly verified as in the classical setting of logarithmic space computability.

Lemma 2.5. *Let κ, κ' be parameterizations and let $F, F' : \{0,1\}^* \to \{0,1\}^*$ be implicitly pl-computable with respect to κ and κ' respectively. Then $F' \circ F$ is implicitly pl-computable with respect to κ.*

Let $(Q, \kappa), (Q', \kappa')$ be parameterized problems. A *pl-reduction* from (Q, κ) to (Q', κ') is a reduction $R : \{0,1\}^* \to \{0,1\}^*$ from Q to Q' that is implicitly pl-computable[2] with respect to κ and such that there exists a computable function $f : \mathbb{N} \to \mathbb{N}$ such that $\kappa' \circ R \leq f \circ \kappa$. We write $(Q, \kappa) \leq_{\text{pl}} (Q', \kappa')$ to indicate that such a reduction exists. We write $(Q, \kappa) \equiv_{\text{pl}} (Q', \kappa')$ if both $(Q, \kappa) \leq_{\text{pl}} (Q', \kappa')$ and $(Q', \kappa') \leq_{\text{pl}} (Q, \kappa)$.

3 Classification

Theorem 3.1 (Classification Theorem). *Let $\mathcal{A}$ be a decidable class of structures of bounded arity such that $\text{core}(\mathcal{A})$ has bounded treewidth.*

1. *If $\text{core}(\mathcal{A})$ has unbounded pathwidth, then*

$$p\text{-HOM}(\mathcal{A}) \equiv_{\text{pl}} p\text{-HOM}(\mathcal{T}^*).$$

2. *If $\text{core}(\mathcal{A})$ has bounded pathwidth and unbounded tree depth, then*

$$p\text{-HOM}(\mathcal{A}) \equiv_{\text{pl}} p\text{-HOM}(\mathcal{P}^*).$$

---

[2] It is routine to verify that F is implicitly pl-computable if and only if it is computable by a pl-space bounded Turing machine with a write-only output tape. Our definition is equivalent to the ones in [15, 6, 12].

3. If $\text{core}(\mathcal{A})$ has bounded tree depth, then

$$p\text{-}\textsc{Hom}(\mathcal{A}) \in \text{para-L}.$$

Remark 3.2. If $\mathcal{A}$ is assumed to be only computably enumerable instead of decidable, then the theorem stays true understanding all mentioned problems in a suitable way as promise problems. If no computability assumption is placed on $\mathcal{A}$, then the theorem stays true in the non-uniform setting of parameterized complexity theory (cf. [11]).

We break the proof into several lemmas.

To prove statement (3) of Theorem 3.1 we show that a structure of *tree depth* w can be characterized, in a sense made precise, by an existential first-order sentence of *quantifier rank* $w+1$, and that model-checking such sentences can be done in parameterized logarithmic space. A proof can be found in Section 3.2.

Lemma 3.3. *Assume $\mathcal{A}$ is a decidable class of structures of bounded arity such that* $\text{core}(\mathcal{A})$ *has bounded tree depth. Then $p\text{-}\textsc{Hom}(\mathcal{A}) \in \text{para-L}$.*

To prove statements (1) and (2) of Theorem 3.1 we need to deal with homomorphism problems for classes $\mathcal{A}$ that are not necessarily decidable. Slightly abusing notation, we say $p\text{-}\textsc{Hom}(\mathcal{A}) \leq_{\text{pl}} p\text{-}\textsc{Hom}(\mathcal{A}')$ for arbitrary classes of structures $\mathcal{A}, \mathcal{A}'$ if there is a implicitly pl-computable *partial* function F that is defined on those instances $(\mathbf{A}, \mathbf{B})$ of $p\text{-}\textsc{Hom}(\mathcal{A})$ with $\mathbf{A} \in \mathcal{A}$ and maps them to equivalent instances $(\mathbf{A}', \mathbf{B}')$ of $p\text{-}\textsc{Hom}(\mathcal{A}')$ with $\mathbf{A}' \in \mathcal{A}'$ such that $|\mathbf{A}'|$ is effectively bounded in $|\mathbf{A}|$. By saying that a partial function F is implicitly pl-computable with respect to a parameterization κ we mean that there are a computable $f : \mathbb{N} \to \mathbb{N}$ and a Turing machine that on those instances (x, i, b) of $\textsc{Bitgraph}(F)$ such that F is defined on x, runs in space $O(f(\kappa(x)) + \log|x|)$ and answers $(x, b, i) \overset{?}{\in} \textsc{Bitgraph}(F)$; on other instances the machine may do whatever it wants.

The following lemma takes care of the reductions from left to right in statements (1) and (2) of Theorem 3.1.

Lemma 3.4. *Let $\mathcal{A}$ be a class of structures and $\mathcal{R} \subseteq \mathcal{T}$ be a computably enumerable class of trees. Assume there is $w \in \mathbb{N}$ such that every structure in $\mathcal{A}$ has a tree decomposition of width at most w whose tree is contained in $\mathcal{R}$. Then,*

$$p\text{-}\textsc{Hom}(\mathcal{A}) \leq_{\text{pl}} p\text{-}\textsc{Hom}(\mathcal{R}^*).$$

Proof. Let $(\mathbf{A}, \mathbf{B})$ with $\mathbf{A} \in \mathcal{A}$ be an instance of $p\text{-}\textsc{Hom}(\mathcal{A})$. Enumerating $\mathcal{R}$, test successively for $\mathbf{T} \in \mathcal{R}$ whether there exists a width $\leq w$ tree-decomposition $(\mathbf{T}, (X_t)_{t \in T})$ of $\mathbf{A}$. Since $\mathbf{A} \in \mathcal{A}$ this test eventually succeeds, and the time needed is effectively bounded in the parameter $|\mathbf{A}|$. With such a tree-decomposition at hand produce the instance $(\mathbf{T}^*, \mathbf{B}')$ of the problem $p\text{-}\textsc{Hom}(\mathcal{R}^*)$ where the structure $\mathbf{B}'$ is defined as follows. Write $\text{dom}(f)$ for the domain of a partial function f; two partial functions f and g are *compatible* if they agree on arguments where they are both

defined.

$$
\begin{aligned}
B' &:= \big\{ f \mid f \text{ is a partial homomorphism from } \mathbf{A} \text{ to } \mathbf{B} \text{ and } |\mathsf{dom}(f)| \le w \big\}; \\
E^{\mathbf{B}'} &:= \big\{ (f,g) \in B' \times B' \mid f \text{ and } g \text{ are compatible} \big\}; \\
C_t^{\mathbf{B}'} &:= \big\{ f \in B' \mid \mathsf{dom}(f) = X_t \big\}, \quad \text{ for every } t \in T.
\end{aligned}
$$

Suppose that h is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$. Then the mapping $h' : T \to B'$ defined by $h'(t) = h \upharpoonright X_t$ is straightforwardly verified to be a homomorphism from $\mathbf{T}^*$ to $\mathbf{B}'$.

Conversely, let h' be a homomorphism from $\mathbf{T}^*$ to $\mathbf{B}'$. Then, $h'(t)$ is a partial homomorphism from $\mathbf{A}$ to $\mathbf{B}$ with domain X_t. Since $\mathbf{T}$ is connected the values of h' are pairwise compatible. Hence $h := \bigcup_{t \in T} h'(t)$ is a function from $\bigcup_{t \in T} X_t = A$ to B. To see h is a homomorphism, consider a tuple $(a_1, \ldots, a_r) \in R^{\mathbf{A}}$ for some r-ary relation R in the vocabulary of $\mathbf{A}$. Then $\{a_1, \ldots, a_r\}$ is contained in some bag X_t since it is a clique in the Gaifman graph of $\mathbf{A}$ (cf. [3, Lemma 4]). But $h'(t)$ maps this tuple to a tuple in $R^{\mathbf{B}}$, so the mapping h does as well. $\square$

For later use we make the following remark concerning the above proof.

Remark 3.5. The previous proof associates with a homomorphism h from $\mathbf{A}$ to $\mathbf{B}$ the homomorphism h' from $\mathbf{T}^*$ to $\mathbf{B}'$ that maps t to $h \upharpoonright X_t$. This association $h \mapsto h'$ is injective because every $a \in A$ appears in some bag X_t. It is also surjective: a homomorphism h' from $\mathbf{T}^*$ to $\mathbf{B}'$, is associated with $h := \bigcup_{t \in T} h'(t)$; the previous proof argued that h is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$. Hence, there is a bijection between the set of homomorphisms from $\mathbf{A}$ to $\mathbf{B}$ and the set of homomorphisms from $\mathbf{T}^*$ to $\mathbf{B}'$.

At the heart of the proof of Theorem 3.1 is the following sequence of reductions, proved in the following subsection. The appropriately informed reader will recognize elements from Grohe's proof [20] as well as from Marx [24, Lemma 5.2].

Lemma 3.6 (Reduction Lemma). *Let $\mathcal{A}$ be a computably enumerable class of structures of bounded arity, let $\mathcal{G}$ be the class of Gaifman graphs of $\mathsf{core}(\mathcal{A})$, and let $\mathcal{M}$ be the class of minors of graphs in $\mathcal{G}$. Then*

$$
\begin{aligned}
p\text{-}\mathrm{HOM}(\mathcal{M}^*) &\le_{\mathrm{pl}} \; p\text{-}\mathrm{HOM}(\mathcal{G}^*) \\
&\le_{\mathrm{pl}} \; p\text{-}\mathrm{HOM}(\mathsf{core}(\mathcal{A})^*) \\
&\le_{\mathrm{pl}} \; p\text{-}\mathrm{HOM}(\mathsf{core}(\mathcal{A})) \\
&\le_{\mathrm{pl}} \; p\text{-}\mathrm{HOM}(\mathcal{A}).
\end{aligned}
$$

With the Reduction Lemma, we can give the proof of the Classification Theorem.

Proof of Theorem 3.1. The reduction from left to right in statements (1) and (2) follow from Lemma 3.4. The reductions from right to left follow from the Reduction Lemma 3.6 via the Excluded Tree Theorem 2.3 (2) and the Excluded Path Theorem 2.3 (3). Statement (3) is proved as Lemma 3.3. $\square$

3.1 Proof of the Reduction Lemma

As a consequence of the assumption that $\mathcal{A}$ is computably enumerable, each of the sets $\mathcal{M}^*$, $\mathcal{G}^*$, $\mathsf{core}(\mathcal{A})^*$, and $\mathsf{core}(\mathcal{A})$ are computably enumerable. The statement of the theorem claims the existence of four reductions. The last one from $p\text{-}\mathrm{HOM}(\mathsf{core}(\mathcal{A}))$ to $p\text{-}\mathrm{HOM}(\mathcal{A})$ is easy to see. We construct the first three in sequence.

Lemma 3.7. *Let $\mathcal{G}$ be a class of graphs which is computable enumerable, and let $\mathcal{M}$ be the class of minors of graphs in $\mathcal{G}$. Then*

$$p\text{-}\mathrm{HOM}(\mathcal{M}^*) \leq_{\mathrm{pl}} p\text{-}\mathrm{HOM}(\mathcal{G}^*).$$

Proof. Let $(\mathbf{M}^*, \mathbf{B})$ with $\mathbf{M}^* \in \mathcal{M}^*$ be an instance of the problem $p\text{-}\mathrm{HOM}(\mathcal{M}^*)$. Enumerating $\mathcal{G}$, test successively for $\mathbf{G} \in \mathcal{G}$ whether $\mathbf{M}$ is a minor of $\mathbf{G}$. Since $\mathbf{M} \in \mathcal{M}$ this test eventually succeeds, and then compute a minor map μ from $\mathbf{M}$ to $\mathbf{G}$. The time needed is effectively bounded in the parameter $|\mathbf{M}^*|$. The reduction then produces the instance $(\mathbf{G}^*, \mathbf{B}')$ of $p\text{-}\mathrm{HOM}(\mathcal{G}^*)$, where $\mathbf{B}'$ is defined as follows. Let I denote the set $\bigcup_{m \in M} \mu(m)$.

$$
\begin{aligned}
B' \quad &:= \quad (M \times B) \,\dot\cup\, \{\bot\}; \\
E^{\mathbf{B}'} \quad &:= \quad \{((m_1, b_1), (m_2, b_2)) \mid [m_1 = m_2 \Rightarrow b_1 = b_2] \text{ and} \\
&\qquad\qquad\qquad\qquad\qquad [(m_1, m_2) \in E^{\mathbf{M}} \Rightarrow (b_1, b_2) \in E^{\mathbf{B}}]\} \\
&\qquad \cup \{(\bot, b') \mid b' \in B'\} \cup \{(b', \bot) \mid b' \in B'\}; \\
C_v^{\mathbf{B}'} \quad &:= \quad \{(m, b) \mid b \in C_m^{\mathbf{B}}\}, \quad \text{if } m \in M \text{ and } v \in \mu(m); \\
C_v^{\mathbf{B}'} \quad &:= \quad \{\bot\}, \quad \text{if } v \notin I.
\end{aligned}
$$

Suppose that h is a homomorphism from $\mathbf{M}^*$ to $\mathbf{B}$. Let $h' : G \to B'$ be the map that sends, for each $m \in M$, the elements in $\mu(m)$ to $(m, h(m))$ and that sends all elements $v \notin I$ to $\bot$. Then h' is a homomorphism from $\mathbf{G}^*$ to $\mathbf{B}'$.

Suppose that g is a homomorphism from $\mathbf{G}^*$ to $\mathbf{B}'$. We show that g is of the form h' for a homomorphism h from $\mathbf{M}^*$ to $\mathbf{B}$. First, by definition of the $C_v^{\mathbf{B}'}$, it holds that $g(v) = \bot$ for all $v \notin I$. Next, let v, w be elements of a set $\mu(m)$, with $m \in M$. The definition of the $C_v^{\mathbf{B}'}$ ensures that $g(v)$ and $g(w)$ have the form $(m, \cdot)$. Since $\mu(m)$ is connected, the definition of $E^{\mathbf{B}'}$ ensures that $g(v) = g(w)$. Finally, suppose that $(m_1, m_2) \in E^{\mathbf{M}}$, let (m_1, b_1) be the image of $\mu(m_1)$ under g, and let (m_2, b_2) be the image of $\mu(m_2)$ under g. We claim that $(b_1, b_2) \in E^{\mathbf{B}}$. But there exist $v_1 \in \mu(m_1)$ and $v_2 \in \mu(m_2)$ such that $(v_1, v_2) \in E^{\mathbf{G}}$. We then have $(g(v_1), g(v_2)) \in E^{\mathbf{B}'}$ and the definition of $E^{\mathbf{B}'}$ ensures that $(b_1, b_2) \in E^{\mathbf{B}}$. $\square$

Lemma 3.8. *Let $\mathcal{A}$ be a computably enumerable class of structures of bounded arity, and let $\mathcal{G}$ be the class of Gaifman graphs of $\mathcal{A}$. Then*

$$p\text{-}\mathrm{HOM}(\mathcal{G}^*) \leq_{\mathrm{pl}} p\text{-}\mathrm{HOM}(\mathcal{A}^*).$$

Proof. Let $(\mathbf{G}^*, \mathbf{B})$ with $\mathbf{G} \in \mathcal{G}$ be an instance of $p\text{-}\mathrm{HOM}(\mathcal{G}^*)$. Similarly as seen in the previous proof, one can compute from $\mathbf{G}$ a structure $\mathbf{A} \in \mathcal{A}$ whose Gaifman

graph is $\mathbf{G}$; in particular, $A = G$ and we write $\mathbf{G} = (A, E^{\mathbf{G}})$. The reduction outputs $(\mathbf{A}^*, \mathbf{B}')$ where $\mathbf{B}'$ is the structure defined as follows.

$$
\begin{aligned}
B' &:= A \times B, \\
C_a^{\mathbf{B}'} &:= \{a\} \times C_a^{\mathbf{B}}, \\
R^{\mathbf{B}'} &:= \Big\{ ((a_1, b_1), \dots (a_{\mathsf{ar}(R)}, b_{\mathsf{ar}(R)})) \in (A \times B)^{\mathsf{ar}(R)} \mid \\
&\qquad \bar{a} \in R^{\mathbf{A}} \text{ and for all } i, j \in [\mathsf{ar}(R)] : \text{ if } a_i \neq a_j, \text{ then } (b_i, b_j) \in E^{\mathbf{B}} \Big\},
\end{aligned}
$$

for $R \in \tau$ where τ denotes the vocabulary of $\mathbf{A}$. We have to show

$$(\mathbf{G}^*, \mathbf{B}) \in p\text{-}\mathrm{HOM}(\mathcal{G}^*) \iff (\mathbf{A}^*, \mathbf{B}') \in p\text{-}\mathrm{HOM}(\mathcal{A}^*).$$

To see this, assume first that h is a homomorphism from $\mathbf{G}^*$ to $\mathbf{B}$. We claim that $h'(a) := (a, h(a))$ defines a homomorphism from $\mathbf{A}^*$ to $\mathbf{B}'$. If $a' \in C_a^{\mathbf{A}^*}$, then $a' = a$ and $h(a') \in C_a^{\mathbf{B}}$ since h is a homomorphism; by definition then $h'(a') = (a, h(a)) \in C_a^{\mathbf{B}'}$. Hence h' preserves the symbols C_a. To show it preserves $R \in \tau$, let $(a_1, \dots, a_{\mathsf{ar}(R)}) \in R^{\mathbf{A}}$. We have to show $((a_1, h(a_1)), \dots, (a_{\mathsf{ar}(R)}, h(a_{\mathsf{ar}(R)}))) \in R^{\mathbf{B}'}$, or equivalently, for all $i, j \in [\mathsf{ar}(R)]$ with $a_i \neq a_j$ that $(h(a_i), h(a_j)) \in E^{\mathbf{B}}$. But if $a_i \neq a_j$, then $(a_i, a_j) \in E^{\mathbf{G}}$ by definition of the Gaifman graph and $(h(a_i), h(a_j)) \in E^{\mathbf{B}}$ follows from h being a homomorphism.

Conversely, assume that h' is a homomorphism from $\mathbf{A}^*$ to $\mathbf{B}'$. By definition of $C_a^{\mathbf{B}'}$ is follows that $h'(a) = (a, h(a))$ for some function $h : A \to B$ such that $h(a) \in C_a^{\mathbf{B}}$. We claim that h is a homomorphism from $\mathbf{G}^*$ to $\mathbf{B}$. It suffices to show $(h(a), h(a')) \in E^{\mathbf{B}}$ whenever $(a, a') \in E^{\mathbf{G}}$. But if $(a, a') \in E^{\mathbf{G}}$, then $a \neq a'$ and there exist $R \in \sigma$ and $(a_1, \dots, a_{\mathsf{ar}(R)}) \in R^{\mathbf{A}}$ and $i, j \in [\mathsf{ar}(R)]$ such that $a = a_i$ and $a' = a_j$. Then $((a_1, h(a_1)), \dots, (a_{\mathsf{ar}(R)}, h(a_{\mathsf{ar}(R)}))) \in R^{\mathbf{B}'}$ because h' is a homomorphism. Since $a_i \neq a_j$ the definition of $E^{\mathbf{B}'}$ implies $(h(a_i), h(a_j)) = (h(a), h(a')) \in E^{\mathbf{B}}$ as desired. $\square$

Recall that the *direct product* $\mathbf{A} \times \mathbf{B}$ of two τ-structures $\mathbf{A}$ and $\mathbf{B}$ has universe $A \times B$ and interprets a relation symbol $R \in \tau$ by $\{((a_1, b_1), \dots, (a_{\mathsf{ar}(R)}, b_{\mathsf{ar}(R)})) \mid \bar{a} \in R^{\mathbf{A}}, \bar{b} \in R^{\mathbf{B}}\}$.

Lemma 3.9. *Let $\mathcal{A}$ be a class of structures. Then*

$$p\text{-}\mathrm{HOM}(\mathsf{core}(\mathcal{A})^*) \leq_{\mathsf{pl}} p\text{-}\mathrm{HOM}(\mathsf{core}(\mathcal{A})).$$

Proof. Let $(\mathbf{D}^*, \mathbf{B})$ with $\mathbf{D} \in \mathsf{core}(\mathcal{A})$ be an instance of $p\text{-}\mathrm{HOM}(\mathsf{core}(\mathcal{A})^*)$. Let $\mathbf{B}_*$ be the restriction of $\mathbf{B}$ to the vocabulary of $\mathbf{D}$. The reduction produces the instance $(\mathbf{D}, \mathbf{B}')$ of the problem $p\text{-}\mathrm{HOM}(\mathsf{core}(\mathcal{A})))$, where

$$\mathbf{B}' := \big\langle \{(d, b) \in D \times B \mid b \in C_d^{\mathbf{B}}\} \big\rangle^{\mathbf{D} \times \mathbf{B}_*}.$$

Suppose that h is a homomorphism from $\mathbf{D}^*$ to $\mathbf{B}$. Then, the mapping $h' : D \to B'$ defined by $h'(d) = (d, h(d))$ is straightforwardly verified to be a homomorphism from $\mathbf{D}$ to $\mathbf{B}'$.

Suppose that g is a homomorphism from $\mathbf{D}$ to $\mathbf{B}'$. Write π_1 and π_2 for the projections that map a pair to its first and second component respectively. The composition $(\pi_1 \circ g)$ is a homomorphism from $\mathbf{D}$ to itself; since $\mathbf{D}$ is a core, $(\pi_1 \circ g)$ is bijective. Hence, there exists a natural $m \geq 1$ such that $(\pi_1 \circ g)^m$ is the identity on D. Define h as $g \circ (\pi_1 \circ g)^{m-1}$. Clearly, h is a homomorphism from $\mathbf{D}$ to $\mathbf{B}'$, so $\pi_2 \circ h$ is a homomorphism from $\mathbf{D}$ to $\mathbf{B}_*$. We claim that $\pi_2 \circ h$ is also a homomorphism from $\mathbf{D}^*$ to $\mathbf{B}$. Observe that $\pi_1 \circ h$ is the identity on D. In other words, for every $d \in D$ there is $b_d \in B$ such that $h(d) = (d, b_d)$. By definition of $\mathbf{B}'$ we get $b_d \in C_d^{\mathbf{B}}$, establishing the claim. $\qquad\square$

Observe that the map h' constructed in the above proof is an embedding. Hence we have the following corollary that we note explicitly for later use.

Corollary 3.10. *Let $\mathcal{A}$ be a class of structures. Then*

$$p\text{-}\mathrm{HOM}(\mathsf{core}(\mathcal{A})^*) \leq_{\mathrm{pl}} p\text{-}\mathrm{EMB}(\mathsf{core}(\mathcal{A})).$$

3.2   Bounded tree depth and para-L

Let τ be a vocabulary. *First-order τ-formulas* are built from *atoms $R\bar{x}, x = x$* by Boolean combinations and existential and universal quantification. Here, $\bar{x}$ is a tuple of variables of length matching the arity of R. We write $\varphi(\bar{x})$ for a (first-order) τ-formula φ to indicate that the free variables in φ are among the components of $\bar{x}$. The *quantifier rank* $\mathsf{qr}(\varphi)$ of a formula φ is defined as follows:

$\mathsf{qr}(\varphi) = 0 \quad$ for atoms φ;

$\mathsf{qr}(\neg\varphi) = \mathsf{qr}(\varphi)$;

$\mathsf{qr}(\varphi \wedge \psi) = \mathsf{qr}(\varphi \vee \psi) = \max\{\mathsf{qr}(\varphi), \mathsf{qr}(\psi)\}$;

$\mathsf{qr}(\exists x \varphi) = \mathsf{qr}(\forall x \varphi) = 1 + \mathsf{qr}(\varphi)$.

The following is standard, but we could not find a reference, so include the simple proof for completeness.

Lemma 3.11. *The parameterized problem*

p-MC(FO)	
Instance:	A structure $\mathbf{A}$, a first-order sentence φ.
Parameter:	$|\varphi|$.
Problem:	$\mathbf{A} \models \varphi$?

can be decided in space $O(|\varphi| \cdot \log|\varphi| + (\mathsf{qr}(\varphi) + \mathsf{ar}(\varphi)) \cdot \log|A|)$, where $\mathsf{qr}(\varphi)$ is the quantifier rank of φ and $\mathsf{ar}(\varphi)$ is the maximal arity over all relation symbols in φ

Proof. We give an algorithm expecting inputs $(\mathbf{A}, \varphi, \alpha)$ where φ is a formula and α is an assignment for φ in $\mathbf{A}$, that is, a map from a superset of the free variables of φ into A. The algorithm determines whether α satisfies φ in $\mathbf{A}$. It executes a depth-first recursion as follows.

If φ is an atom $R\bar{y}$ the algorithm writes the tuple $\alpha(\bar{y}) \in A^{\mathsf{ar}(R)}$ on the worktape and checks whether it is contained in $R^{\mathbf{A}}$ by scanning the input; it then erases the tuple and returns the bit corresponding to the answer obtained.

If $\varphi = (\psi \wedge \chi)$, the algorithm recurses on ψ (with the same assignment); upon completing the recursion it erases all space used in it, stores a bit for the answer obtained, and then recurses on χ; upon completion it erases the space used in it and returns the minimum of the bit obtained and the stored bit. The cases $\varphi = (\psi \vee \chi)$ and $\varphi = \neg \psi$ are similar.

If $\varphi(\bar{x}) = \exists y \psi(\bar{x}, y)$ the algorithm loops through $b \in A$ and recurses on ψ with assignment α extended by mapping y to b; it maintains a bit which is intially 0 and updates it after each loop to the maximum of the bit obtained in the loop; after each loop it erases the space used in in it. Upon completing the loop it returns this bit, and restricts the assignment back to its old domain without y. The case $\varphi(\bar{x}) = \forall y \psi(\bar{x}, y)$ is similar.

When started on a sentence φ and the empty assignment, all assignments α occuring in the recursion have cardinality $\leq \mathsf{qr}(\varphi)$, so can be stored in space $O(\mathsf{qr}(\varphi) \cdot (\log |\varphi| + \log |A|))$. Each recursive step adds space $O(\log |\varphi|)$ to remember the (position of) the current subformula plus one bit plus $O(\log |A|)$ for the loop on $b \in A$ in the quantifier case and plus $O(\mathsf{ar}(\varphi) \cdot \log |A|)$ in the atomic case. From these considerations it is routine to verify the claimed upper bound on space. $\qquad \square$

The *canonical conjunction* of a structure $\mathbf{A}$ is a quantifier-free conjunction in the variables x_a for $a \in A$; namely, for every relation symbol R of $\mathbf{A}$ and every $(a_1, \ldots, a_{\mathsf{ar}(R)}) \in R^{\mathbf{A}}$ it contains the conjunct $Rx_{a_1} \cdots x_{a_{\mathsf{ar}(R)}}$. It is easy to see that the canonical conjunction of $\mathbf{A}$ is satisfiable in a structure $\mathbf{B}$ if and only if there is an homomorphism from $\mathbf{A}$ to $\mathbf{B}$.

Proof of Lemma 3.3. Choose $w \in \mathbb{N}$ such that $\mathsf{td}(\mathsf{core}(\mathbf{A})) \leq w$ for all $\mathbf{A} \in \mathcal{A}$. Given a structure $\mathbf{A}$ we compute a sentence $\varphi_{\mathbf{A}}$ of quantifier rank at most $w + 1$ such that for all structures $\mathbf{B}$, the sentence $\varphi_{\mathbf{A}}$ is true in $\mathbf{B}$ if and only if there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$. This is enough by Lemma 3.11.

Given $\mathbf{A}$ we check $\mathbf{A} \in \mathcal{A}$ running some decision procedure for $\mathcal{A}$. If $\mathbf{A} \notin \mathcal{A}$ we let $\varphi_{\mathbf{A}} := \exists x \, \neg x = x$. If $\mathbf{A} \in \mathcal{A}$, compute the core $\mathbf{A}_0$ of $\mathbf{A}$ and compute for every connected component C of the Gaifman graph of $\mathbf{A}_0$ some rooted tree $\mathbf{T}$ with vertices $T = C$ and height at most w such that every edge of the Gaifman graph of $\langle C \rangle^{\mathbf{A}_0}$ is in the closure of $\mathbf{T}$.

Consider a component C and let $\mathbf{T}$ be the rooted tree computed for C. For $c \in C = T$ we compute the following first-order formula φ_c. We use variables x_c for $c \in C = T$. If c is a leaf of $\mathbf{T}$, let φ_c be the canonical conjunction of $\langle P_c \rangle^{\mathbf{A}_0}$ where P_c is the path in $\mathbf{T}$ leading from the root r of $\mathbf{T}$ to c. For an inner vertex c define

$$\varphi_c := \bigwedge\nolimits_d \exists x_d \, \varphi_d,$$

where d ranges over the successors of c. The following claims are straightforwardly verified by induction along the recursive definition of the φ_cs.

Claims. For every $c \in C$:

1. the quantifier rank of φ_c equals the height of the subtree of $\mathbf{T}$ rooted at c;

2. the free variables of φ_c are $\{x_d \mid d \in P_c\}$;

3. φ_c is satisfiable in $\mathbf{B}$ if and only if so is the canonical conjunction of $\langle C(c) \rangle^{\mathbf{A}_0}$ where $C(c)$ contains P_c and the vertices in the subtree rooted at c.

Letting r range over the roots of the trees $\mathbf{T}$ chosen for the connected components C of $\mathbf{A}_0$, we set

$$\varphi_{\mathbf{A}} := \bigwedge_r \exists x_r \varphi_r.$$

By Claim 2 this is a sentence and by Claim 1 it has quantifier rank at most $w + 1$. It is true in $\mathbf{B}$ if and only if every $\exists x_r \varphi_r$ is true in $\mathbf{B}$, and by Claim 3 this holds if and only if the canonical conjunction of $\langle C(r) \rangle^{\mathbf{A}_0}$ is satisfiable in $\mathbf{B}$ for every connected component C. Noting $C(r) = C$, this means that every $\langle C \rangle^{\mathbf{A}_0}$ maps homomorphically to $\mathbf{B}$, and this means that $\mathbf{A}_0$ maps homomorphically to $\mathbf{B}$. Recalling that $\mathbf{A}_0$ is the core of $\mathbf{A}$, we see that this is equivalent to $\mathbf{A}$ mapping homomorphically to $\mathbf{B}$. $\square$

Define a $\{\wedge, \exists\}$-sentence to be a first-order sentence built from atoms, conjunction, and existential quantification. The previous proof revealed that, given a structure $\mathbf{A}$ with $\mathsf{td}(\mathsf{core}(\mathbf{A})) \leq w$, there exists a $\{\wedge, \exists\}$-sentence ϕ of quantifier rank at most $w + 1$ that *corresponds* to $\mathbf{A}$ in that, for all structures $\mathbf{B}$, the sentence ϕ is true on $\mathbf{B}$ if and only if there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$. We show that the existence of such a sentence in fact characterizes tree depth, in the following precise sense.

Theorem 3.12. *Let $w \geq 0$, and let $\mathbf{A}$ be a structure. It holds that $\mathsf{td}(\mathsf{core}(\mathbf{A})) \leq w$ if and only if there exists a $\{\wedge, \exists\}$-sentence ϕ that corresponds to $\mathbf{A}$ with $\mathsf{qr}(\phi) \leq w + 1$.*

Proof. The forward direction follows from the previous proof. For the backward direction, let ϕ be a sentence of the described type. We may assume that no variable is quantified twice in ϕ and that no equality of variables appears in ϕ, by renaming variables and replacing equalities of the form $v = v$ with the empty conjunction. Let ϕ_p be the prenex sentence where all variables that are existentially quantified in ϕ are existentially quantified in ϕ_p, and the quantifier-free part of ϕ_p is the conjunction of all atoms appearing in ϕ. Let $\mathbf{C}$ be a structure whose canonical conjunction is the quantifier-free part of ϕ_p. Clearly, ϕ_p and the original ϕ are logically equivalent; it follows that $\mathbf{C}$ and $\mathbf{A}$ are homomorphically equivalent [4]. It thus suffices to show that $\mathsf{td}(\mathbf{C}) \leq w$.

View the sentence ϕ as a directed graph, and define an acyclic directed graph D on the variables of ϕ where the directed edge (v, v') is present if and only if the node for $\exists v$ is the first node with quantification occurring above the node for $\exists v'$. Let α be an arbitrary atom from ϕ_p (equivalently, from ϕ). Since ϕ is a sentence, if one traverses ϕ starting from the root and moving to α, one will pass a node $\exists v$ for each variable v of α. Let $v_1, \ldots, v_k$ be the variables of α in the order encountered by such a traversal. The edges $(v_1, v_2), (v_2, v_3), \ldots, (v_{k-1}, v_k)$ are in the transitive closure of D, and hence in the closure of the graph underlying D (where a node is a root in the graph iff it is parentless in D). Since $\mathsf{qr}(\phi) \leq w + 1$, each directed path in D has length less than or equal to w, and so the graph underlying D witnesses that $\mathsf{td}(\mathbf{C}) \leq w$. $\square$

We now show the following result on the embedding problem.

Theorem 3.13. *Assume $\mathcal{A}$ is a decidable class of structures of bounded arity and bounded tree depth. Then p-EMB$(\mathcal{A}) \in$ para-L.*

The proof of this result uses color coding methods, more precisely, it relies on the following lemma (see [17, p.349]).

Lemma 3.14. *For every sufficiently large n, it holds that for all $k \in \mathbb{N}$ and for every k-element subset X of $[n]$, there exists a prime $p < k^2 \log n$ and $q < p$ such that the function $h_{p,q} : [n] \to \{0, \ldots, k^2 - 1\}$ given by*

$$h_{p,q}(m) := (q \cdot m \bmod p) \bmod k^2$$

is injective on X.

For later use we give the main step in the proof of Theorem 3.13 as a separate lemma. Call a structure *connected* if its Gaifman graph is connected.

Lemma 3.15. *For every decidable class of connected structures $\mathcal{A}$ we have*

$$p\text{-EMB}(\mathcal{A}) \leq_{\text{pl}} p\text{-HOM}(\mathcal{A}^*).$$

Proof. Map an instance $(\mathbf{A}, \mathbf{B})$ to $(\mathbf{A}^*, \mathbf{B}_*)$ where $\mathbf{B}_*$ is defined as follows. We assume that $B = [|B|]$ and $A = [|A|]$. Let F be the set

$$\left\{ g \circ h_{p,q} \mid g : \{0, \ldots, |A|^2 - 1\} \to A \text{ and } q < p < |A|^2 \log |B| \right\}.$$

Here, $h_{p,q} : [|B|] \to \{0, \ldots, |A|^2 - 1\}$ is the function from Lemma 3.14 (for $n := |B|$ and $k := |A|$). For $f \in F$, let $\mathbf{B}_f$ be the expansion of $\mathbf{B}$ that interprets every $C_a, a \in A$, by $f^{-1}(a) \subseteq B$ and define $\mathbf{B}_*$ as the disjoint union of the structures $\mathbf{B}_f$. We verify

$$(\mathbf{A}, \mathbf{B}) \in p\text{-EMB}(\mathcal{A}) \Longleftrightarrow (\mathbf{A}^*, \mathbf{B}_*) \in p\text{-HOM}(\mathcal{A}^*).$$

Note that the sets $C_a^{\mathbf{B}_*}, a \in A$, are pairwise disjoint, so every homomorphism from $\mathbf{A}^*$ to $\mathbf{B}_*$ is an embedding. And because $\mathbf{A}^*$ is connected, it is an embedding into (the copy of) some $\mathbf{B}_f$, so it corresponds to an embedding from $\mathbf{A}$ into $\mathbf{B}$. Conversely, assume e is an embedding of $\mathbf{A}$ into $\mathbf{B}$. By Lemma 3.14 there are p, q with $q < p < |A|^2 \log |B|$ such that $h_{p,q}$ is injective on the image of e. Then there exists $g : \{0, \ldots, |A|^2 - 1\} \to A$ such that $g \circ h_{p,q} \circ e$ is the identity on A. Then $f := g \circ h_{p,q} \in F$ and e is an embedding of $\mathbf{A}^*$ into $\mathbf{B}_f$ and hence into $\mathbf{B}_*$. $\square$

This lemma together with Corollary 3.10 implies:

Corollary 3.16. *Let $\mathcal{A}$ be a decidable class of connected cores. Then*

$$p\text{-HOM}(\mathcal{A}^*) \equiv_{\text{pl}} p\text{-EMB}(\mathcal{A}).$$

Proof of Theorem 3.13. Let $\mathcal{A}$ accord the assumption.

Claim. There exists a decidable class of connected structures $\mathcal{A}'$ of bounded tree depth such that p-EMB$(\mathcal{A}) \leq_{\text{pl}} p$-EMB$(\mathcal{A}')$.

Note $p\text{-}\textsc{Emb}(\mathcal{A}') \leq_{\mathrm{pl}} p\text{-}\textsc{Hom}((\mathcal{A}')^*)$ by the previous lemma and $p\text{-}\textsc{Hom}((\mathcal{A}')^*) \in$ para-L by Lemma 3.3. We are thus left to prove the claim.

Assume $\mathcal{A}$ has tree depth at most d and let E be a binary relation symbol not occuring in the vocabulary of any $\mathbf{A} \in \mathcal{A}$. Fix a computable function that maps every $\mathbf{A} \in \mathcal{A}$ to a family of height $\leq d$ rooted trees $(\mathbf{T}_C)_C$ with $\mathbf{T}_C = (C, E^{\mathbf{T}_C}, root^{\mathbf{T}_C})$ where C ranges over the connected components of the Gaifman graph $\mathbf{G}(\mathbf{A})$ of $\mathbf{A}$, and such that $\langle C \rangle^{\mathbf{G}(\mathbf{A})}$ is a subgraph of the closure of $\mathbf{T}_C$. Define $\mathbf{A}'$ to be the expansion of $\mathbf{A}$ interpreting E by $\bigcup_C E^{\mathbf{T}_C} \cup E'$ where E' is defined as follows. It contains edges between the root of $\mathbf{T}_{C_0}$ and the roots of the other $\mathbf{T}_C$ where C_0 is the lexicographically minimal component (according to the encoding of $\mathbf{A}$). Then $\mathbf{A}'$ is connected and has tree depth at most $d + 1$. Clearly, $\mathcal{A}' := \{\mathbf{A}' \mid \mathbf{A} \in \mathcal{A}\}$ is decidable. The map $(\mathbf{A}, \mathbf{B}) \mapsto (\mathbf{A}', \mathbf{B}')$, where $\mathbf{B}'$ is the expansion of $\mathbf{B}$ interpreting E by B^2, is a pl-reduction from $p\text{-}\textsc{Emb}(\mathcal{A})$ to $p\text{-}\textsc{Emb}(\mathcal{A}')$. $\qquad\square$

4 The class PATH

We present the complexity class PATH to capture the complexity of $p\text{-}\textsc{Hom}(\mathcal{P}^*)$. This class was discovered very recently by Elberfeld et al. [12] with a different angle of motivation; they refer to this class as para-NL$[f\log]$. Among other results, they show that the following problem is complete for this class: check if a digraph contains a path from a distinguished vertex s to another distinguished vertex t of length at most k; here, k is the parameter. We use $p\text{-}st\text{-}\textsc{Path}$ to denote the corresponding problem for undirected graphs.

$p\text{-}st\text{-}\textsc{Path}$	
Instance:	A graph $\mathbf{G}$, $s, t \in G$ and $k \in \mathbb{N}$.
Parameter:	k.
Problem:	Is there a path in $\mathbf{G}$ from s to t of length at most k ?

Definition 4.1. The class PATH contains a parameterized problem (Q, κ) if there are a computable function $f : \mathbb{N} \to \mathbb{N}$ and a nondeterministic Turing machine that accepts Q, is pl-space bounded with respect to κ, and uses $f(\kappa(x)) \cdot \log |x|$ many nondeterministic bits.

The following is straightforward to verify.

Proposition 4.2. *The complexity class* PATH *is closed under pl-reductions.*

Recall that, using the notation in [15], one has

$$\text{FPT} = \text{para-P} \subseteq \text{W[P]} \subseteq \text{para-NP}.$$

It follows immediately from the definitions that

$$\text{para-L} \subseteq \text{PATH} \subseteq \text{para-NL}.$$

The class PATH is natural in that it has a natural machine characterization that is analogous to the one of W[P]. We shall see that it captures the complexity of many natural problems.

Theorem 4.3. p-HOM$(\mathcal{P}^*)$ *is complete for* PATH *under pl-reductions.*

That p-HOM$(\mathcal{P}^*)$ is contained in PATH can be seen by the guess-and-check paradigm. We find it informative to present such algorithms in a computational model tailored specifically for this kind of nondeterminism.

Definition 4.4. A *jump machine* is a Turing machine with an input tape and a special *jump state*. When the machine enters the jump state the head on the input tape is set nondeterministically on one of the cells carrying an input bit; we say that the machine *jumps to* the cell. When this occurs, no other head moves or writes and the state is changed to the starting state. Acceptance is defined as usual, that is, such a machine accepts an input if there exists a sequence of nondeterministic jump choices under which the machine accepts. An *injective jump machine* is defined similarly to a jump machine, but never jumps to a cell that has already been jumped to.

For a function $j : \{0,1\}^* \to \mathbb{N}$, we say that a jump machine (an injective jump machine) uses j *many* (injective) jumps if for every input x and every run on x, it enters the jump state at most $j(x)$ many times.

The idea is that a jump corresponds to a guess of a number in $[n]$ where n is the length of the input. Observe that one can compute in logarithmic space the number $m \in [n]$ of the cell it jumps to by moving the head to the left and stepwise increasing a counter.

Lemma 4.5. *Let* (Q, κ) *be a parameterized problem. The following are equivalent.*

1. $(Q, \kappa) \in$ PATH.

2. *There exists a computable* $f : \mathbb{N} \to \mathbb{N}$ *and a jump machine* $\mathbb{A}$ *using* $(f \circ \kappa)$ *many jumps that accepts* Q *and is pl-space bounded with respect to* κ.

3. *There exists a computable* $f : \mathbb{N} \to \mathbb{N}$ *and an injective jump machine* $\mathbb{A}$ *using* $(f \circ \kappa)$ *many injective jumps that accepts* Q *and is pl-space bounded with respect to* κ.

Proof. (1) implies (2): assume (1) and choose $\mathbb{A}$ and f according Definition 4.1. Given an input x we simulate $\mathbb{A}$ by a jump machine $\mathbb{B}$ that makes use of an extra worktape. When $\mathbb{A}$ enters its guess state $\mathbb{B}$ moves its head on the extra worktape right and continues the simulation of $\mathbb{A}$ in state s_b where $b \in \{0, 1\}$ is the bit scanned by this head. In case the head scans a blank cell, $\mathbb{B}$ stores the number j of the cell its input head is scanning and then performs a jump, say to cell $m \in [|x|]$. It computes the binary code of m of length $\lceil \log(|x| + 1) \rceil$. It overwrites the content of the extra worktape by this code and sets its head on the first bit b of the code, moves the input head back to cell j and continues the simulation of $\mathbb{A}$ in state s_b. Then $\mathbb{B}$ makes at most $f(\kappa(x))$ many jumps.

(2) implies (3): let $\mathbb{A}$ and f accord (2). To get a machine according to (3) we intend to simply simulate $\mathbb{A}$ on an injective jump machine. This works provided $\mathbb{A}$ does not have accepting runs with two jumps to the same cell. To ensure this condition we replace $\mathbb{A}$ by the following machine $\mathbb{A}'$. Intuitively, if $\mathbb{A}$ jumps k times then $\mathbb{A}'$ jumps $2k$ times and accepts only if these $2k$ jumps encode pairs $(1, m_1), \ldots, (2k, m_{2k})$; the

simulation of the ith jump of $\mathbb{A}$ is done by jumping to the (m_{2i}, m_{2i+1})th cell. Details follow.

The machine $\mathbb{A}'$ on x first computes $k := f(\kappa(x))$: note $\kappa(x)$ can be computed in space $O(\log |x|)$ by our convention on parameterizations; then k can be computed from $\kappa(x)$ running some machine computing f on $\kappa(x)$ – this needs additional space which is effectively bounded in the parameter $\kappa(x)$.

Then $\mathbb{A}'$ checks that $2k \cdot \lceil \sqrt{n} \rceil \leq n$ where $n := |x|$. If this check fails, $\mathbb{A}'$ simulates some fixed decision procedure for Q (note that (2) implies that Q is decidable). Observe that in this case $k \geq \Omega(\sqrt{n})$, so the decision procedure runs in space effectively bounded in k and hence in the parameter. Otherwise $2k \cdot \lceil \sqrt{n} \rceil \leq n$ and $\mathbb{A}'$ simulates $\mathbb{A}$ as follows. Throughout the simulation it maintains a counter for jumps that initially is set to 0. It will be clear that this counter always stores a number $\leq 2k$.

When $\mathbb{A}$ jumps, $\mathbb{A}'$ jumps twice and computes the two numbers a, b of the cells it jumped to. It interprets a, b as encoding pairs $(i_a, m_a), (i_b, m_b) \in [2k] \times [\lceil \sqrt{n} \rceil]$. More precisely, $i_a := \lceil a/\lceil \sqrt{n} \rceil \rceil$ is the least i such that $i \cdot \lceil \sqrt{n} \rceil \geq a$ and $m_a := 1 + a - (i_a - 1) \cdot \lceil \sqrt{n} \rceil$; similarly for (i_b, m_b). If (i_a, m_a) or (i_b, m_b) is not in $[2k] \times [\lceil \sqrt{n} \rceil]$, then $\mathbb{A}'$ halts and rejects.

For i the value of the jump counter, $\mathbb{A}'$ checks that $i + 1 = i_a$ and that $i + 2 = i_b$. Then it computes $m := m_a \cdot \lceil \sqrt{n} \rceil + m_b$ and checks that $m \in [n]$. Then $\mathbb{A}'$ increases the jump counter by two, moves the input head to cell m, changes to the starting state and resumes the simulation of $\mathbb{A}$.

(3) implies (1): choose a machine $\mathbb{A}$ and a function f according (3) and define a machine $\mathbb{B}$ as follows. On x it first computes $k := f(\kappa(x))$ (within allowed space as seen above) and $n := |x|$. If $k \geq \log n$ it runs some fixed machine $\mathbb{Q}$ deciding Q and answers accordingly. Since $k \geq \log n$ this needs space effectively bounded in k and thus in the parameter. If otherwise $k < \log n$, then $\mathbb{B}$ simulates $\mathbb{A}$ as follows. During the simulation it maintains a set X containing at most k natural numbers all smaller k^2 – intuitively, this set contains fingerprints of the jumps sofar. Initially, $X = \emptyset$.

To begin, $\mathbb{B}$ guesses a pair (p, q) with $q < p < k^2 \log n$ and stores it. Note that this requires only $O(\log k + \log \log n) \leq O(\log \log n)$ nondeterministic bits and space. Then $\mathbb{B}$ starts simulating $\mathbb{A}$. When $\mathbb{A}$ jumps, $\mathbb{B}$ guesses $\lceil \log(n+1) \rceil$ many bits encoding a number $m \in [n]$. It computes $f := h_{p,q}(m)$ and checks that $f \notin X$. Then it adds f to X, moves the input head to the mth input bit, changes to the starting state and continues the simulation of $\mathbb{A}$.

Obviously, if $\mathbb{A}$ jumps at most ℓ times, then $\mathbb{B}$ uses at most $O(\log \log n + \ell \log n)$ nondeterministic bits. To see that $\mathbb{B}$ runs in allowed space, observe that the "fingerprint" f can be computed in space $O(\log n)$: first $b := qm \bmod p$ can trivially be computed in space polynomial in $\log p$ and this is space $(\log \log n)^{O(1)} \leq O(\log n)$; second, $f = b \bmod k^2$ can trivially be computed in space polynomial in $(\log k + \log b)$ and the space usage her is $(\log \log n)^{O(1)}$.

We show that $\mathbb{B}$ accepts x if and only if $x \in Q$. If $\mathbb{B}$ accepts x then either because $\mathbb{Q}$ accepts x (and then trivially $x \in Q$) or because $\mathbb{A}$ reaches an accepting state when it jumps to cells numbered $m_1, \ldots, m_\ell$; note that the fingerprints of these cell numbers are pairwise different, and hence so are the numbers. This implies $x \in Q$. Conversely, if $x \in Q$, then there is an accepting run of $\mathbb{A}$ on x with $\ell \leq k$ jumps to pairwise different cells $m_1, \ldots, m_\ell$. By Lemma 3.14 there exist $q < p < k^2 \log n$ such that

$h_{p,q}$ is injective on $\{m_1, \ldots, m_\ell\}$. Then $\mathbb{B}$ accepts when first guessing some such pair (p, q) and then strings encoding $m_1, \ldots, m_\ell$. $\square$

Theorem 4.6. *Let $\mathcal{A}$ be a decidable class of structures of bounded arity and of bounded pathwidth. Then $p\text{-}\textsc{Emb}(\mathcal{A}) \in \text{PATH}$.*

Proof. Choose a constant $w \in \mathbb{N}$ bounding the pathwidth of $\mathcal{A}$. We use a machine $\mathbb{A}$ with injective jumps to solve $p\text{-}\textsc{Emb}(\mathcal{A})$. The result will then follow from Lemma 4.5.

Given an instance $(\mathbf{A}, \mathbf{B})$ of $p\text{-}\textsc{Emb}(\mathcal{A})$ the machine first computes a width $\leq w$ path-decomposition $(\mathbf{P}_k, (X_i)_{i \in [k]})$ of $\mathbf{A}$ such that $X_i \subsetneq X_{i+1}$ or $X_{i+1} \subsetneq X_i$ for all $i \in [k-1]$; we further assume that no X_i is empty. This is done in space effectively bounded in the parameter $|\mathbf{A}|$ and, in particular, k is effectively bounded in $|\mathbf{A}|$.

It then computes inductively for each $i \in [k]$ a map h_i from X_i into B that is a partial homomorphism from $\mathbf{A}$ into $\mathbf{B}$. To start, the machine $\mathbb{A}$ jumps $|X_1|$ times to guess elements $b_1, \ldots b_{|X_1|} \in B$. It checks that the function $h_1 : X_1 \to B$ that maps the ith element of X_1 to b_i defines a partial homomorphism from $\mathbf{A}$ into $\mathbf{B}$. Having computed h_i the machine computes h_{i+1} as follows. If $X_{i+1} \subsetneq X_i$, then $h_{i+1} := h_i \upharpoonright X_{i+1}$ is the restriction of h_i to X_{i+1}. Otherwise $X_{i+1} \supsetneq X_i$, say $X_{i+1} = X_i \cup \{a_1, \ldots, a_d\}$; then $\mathbb{A}$ jumps d times to guess $b_1, \ldots b_d \in B$ and checks that $h_{i+1} := (h_i \upharpoonright X_i) \cup \{(a_j, b_j) \mid j \in [d]\}$ is a partial homomorphism from $\mathbf{A}$ into $\mathbf{B}$. In the end, if no check fails, $\mathbb{A}$ halts accepting.

This procedure can be implemented in pl-space: the space to store the path de-composition is bounded in the parameter, and storing one h_i needs space roughly $w \cdot (\log |A| + \log |B|)$.

It is routine to check that $\mathbb{A}$ makes exactly $|A|$ many jumps, and that it accepts only if $\bigcup_i h_i$ is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$. Since the machine has injective jumps it accepts in fact only if this homomorphism is an embedding. Conversely, it is obvious that the machine accepts if an embedding from $\mathbf{A}$ into $\mathbf{B}$ exists. $\square$

Proof of Theorem 4.3. To see $p\text{-}\textsc{Hom}(\mathcal{P}^*) \in \text{PATH}$, just consider the machine $\mathbb{A}$ de-scribed in the proof of Theorem 4.6 as a machine with jumps instead of as a machine with injective jumps.

To see that $p\text{-}\textsc{Hom}(\mathcal{P}^*)$ is hard for PATH under pl-reductions, let $(Q, \kappa) \in \text{PATH}$ and choose a Turing machine $\mathbb{A}$ with jumps according Lemma 4.5 (2) that accepts Q. We can assume that there are computable $f, g : \mathbb{N} \to \mathbb{N}$ such that $\mathbb{A}$ on $x \in \{0,1\}^*$ runs in space $O(g(\kappa(x)) + \log |x|)$ and makes on every run exactly $f(\kappa(x))$ many jumps.

Fix $x \in \{0,1\}^*$ and set $k := \kappa(x)$ and $n := |x|$. Let $\mathbb{A}_{det}$ be the deterministic Turing machine defined as $\mathbb{A}$ but with the jump state interpreted as a rejecting halting state. Observe that $\mathbb{A}_{det}$ (and $\mathbb{A}$) has at most $m := 2^{g(k)} \cdot n^c$ configurations where $c \in \mathbb{N}$ is a suitable constant. Let $c_1, \ldots, c_m$ be a list (possibly with repetitions) of all configurations of $\mathbb{A}_{det}$ on x whose state is the starting state. Assume that c_1 is the starting configuration of $\mathbb{A}_{det}$. For $i, j \in [m]$, say i *reaches* j if the computation of $\mathbb{A}_{det}$ started on c_i (with x on the input tape) reaches in at most m steps a configuration c with the jump state, and c_j is obtained from c by changing the jump state to the starting state and changing the position of the input head to some arbitrary cell storing an input bit. Further, call $i \in [m]$ *accepting* if $\mathbb{A}_{det}$ started on c_i accepts within at most m steps.

Consider the structure $\mathbf{B}_x$ given by

$$
\begin{aligned}
B_x &:= [f(k)+1] \times [m], \\
E^{\mathbf{B}_x} &:= \text{the symmetric closure of} \\
&\qquad \{((i,j),(i+1,j')) \mid i \in [f(k)], j \text{ reaches } j'\}, \\
C_1^{\mathbf{B}_x} &:= \{(1,1)\}, \\
C_i^{\mathbf{B}_x} &:= \{i\} \times [m] \text{ for } 2 \le i \le f(k), \\
C_{f(k)+1}^{\mathbf{B}_x} &:= \{(f(k)+1,j) \mid j \text{ is accepting}\}.
\end{aligned}
$$

It is clear that there exists a homomorphism from $\mathbf{P}_{f(k)+1}^*$ to $\mathbf{B}_x$ if and only if $\mathbb{A}$ accepts x, that is, the map $x \mapsto (\mathbf{P}_{f(\kappa(x))+1}^*, \mathbf{B}_x)$ is a reduction from (Q, κ) to $p\text{-}\mathrm{HOM}(\mathcal{P}^*)$. The new parameter $|\mathbf{P}_{f(\kappa(x))+1}^*|$ depends only on $\kappa(x)$. The reduction is implicitly pl-computable: first observe that the numbers $f(k)$ and m can be computed from x in pl-space. A counter for numbers up to m needs only space $O(g(k) + \log n)$. Hence one can tell whether or not i reaches j in pl-space simply by simulating $\mathbb{A}_{det}$ for at most m many steps. Similarly, this space is sufficient to tell whether or not a given $j \in [m]$ is accepting. $\square$

The following result gives information about fundamental problems: the problems $p\text{-}\mathrm{EMB}(\overrightarrow{\mathcal{P}})$, $p\text{-}\mathrm{EMB}(\mathcal{C})$, and $p\text{-}\mathrm{EMB}(\overrightarrow{\mathcal{C}})$ are the parameterized problems of determining if an input graph contains a simple directed k-path, a simple undirected k-cycle, and a simple directed k-cycle, respectively; these problems are denoted respectively by $p\text{-}\mathrm{DIRPATH}$, $p\text{-}\mathrm{CYCLE}$, and $p\text{-}\mathrm{DIRCYCLE}$ by Flum and Grohe [16].

Theorem 4.7. *The following parameterized problems are complete for* PATH *under pl-reductions:*

$$
\begin{aligned}
&p\text{-}st\text{-}\mathrm{PATH}, \\
&p\text{-}\mathrm{HOM}(\overrightarrow{\mathcal{P}}), \quad p\text{-}\mathrm{EMB}(\overrightarrow{\mathcal{P}}) \\
&p\text{-}\mathrm{HOM}(\mathcal{C}), \quad p\text{-}\mathrm{EMB}(\mathcal{C}) \\
&p\text{-}\mathrm{HOM}(\overrightarrow{\mathcal{C}}), \quad p\text{-}\mathrm{EMB}(\overrightarrow{\mathcal{C}})
\end{aligned}
$$

Proof. By Theorem 4.6 all embedding problems are contained in PATH. For the homomorphism problems and $p\text{-}st\text{-}\mathrm{PATH}$ the same argument works (see the proof of Theorem 4.3). We are thus left to prove hardness.

Recall Example 2.1. Corollary 3.10 implies that $p\text{-}\mathrm{HOM}(\overrightarrow{\mathcal{P}}^*) \le_{\mathrm{pl}} p\text{-}\mathrm{EMB}(\overrightarrow{\mathcal{P}})$ and also that $p\text{-}\mathrm{HOM}(\overrightarrow{\mathcal{C}}^*) \le_{\mathrm{pl}} p\text{-}\mathrm{EMB}(\overrightarrow{\mathcal{C}})$. Since we trivially have $p\text{-}\mathrm{HOM}(\mathcal{A}) \le_{\mathrm{pl}} p\text{-}\mathrm{HOM}(\mathcal{A}^*)$ for all classes $\mathcal{A}$, we conclude that $p\text{-}\mathrm{HOM}(\overrightarrow{\mathcal{P}}) \le_{\mathrm{pl}} p\text{-}\mathrm{EMB}(\overrightarrow{\mathcal{P}})$ and also that $p\text{-}\mathrm{HOM}(\overrightarrow{\mathcal{C}}) \le_{\mathrm{pl}} p\text{-}\mathrm{EMB}(\overrightarrow{\mathcal{C}})$. For $\mathcal{C}$ we similarly get $p\text{-}\mathrm{HOM}(\mathcal{C}_{\mathrm{odd}}) \le_{\mathrm{pl}} p\text{-}\mathrm{EMB}(\mathcal{C}_{\mathrm{odd}})$ where $\mathcal{C}_{\mathrm{odd}}$ is the class of odd length cycles. By $\overrightarrow{\mathcal{C}}_{\mathrm{odd}}$ we denote the class of odd length directed cycles.

It thus suffices to show that the problems

$$
p\text{-}\mathrm{HOM}(\overrightarrow{\mathcal{P}}), p\text{-}\mathrm{HOM}(\overrightarrow{\mathcal{C}}), p\text{-}\mathrm{HOM}(\mathcal{C}_{\mathrm{odd}}), p\text{-}st\text{-}\mathrm{PATH}
$$

are PATH-hard. By Theorem 4.3, we know that p-HOM$(\mathcal{P}^*)$ is hard for PATH. We give the sequence of reductions

$$p\text{-HOM}(\mathcal{P}^*) \leq_{\mathrm{pl}} p\text{-HOM}(\overrightarrow{\mathcal{P}}) \leq_{\mathrm{pl}} p\text{-}st\text{-PATH} \leq_{\mathrm{pl}} p\text{-HOM}(\overrightarrow{\mathcal{C}}_{\mathrm{odd}})$$

and then show the hardness of p-HOM$(\mathcal{C}_{\mathrm{odd}})$.

p-HOM$(\mathcal{P}^*) \leq_{\mathrm{pl}} p$-HOM$(\overrightarrow{\mathcal{P}})$. Let $(\mathbf{P}_k^*, \mathbf{B})$ be an instance of p-HOM$(\mathcal{P}^*)$. The reduction produces the instance $(\overrightarrow{\mathbf{P}_k}, \mathbf{B}')$ where $\mathbf{B}'$ is the directed graph with vertices $B' := [k] \times B$ and edges

$$E^{\mathbf{B}'} := \{((i,b),(i+1,b')) \mid i \in [k-1], b \in C_i^{\mathbf{B}}, b' \in C_{i+1}^{\mathbf{B}}\}.$$

p-HOM$(\overrightarrow{\mathcal{P}}) \leq_{\mathrm{pl}} p$-$st$-PATH. Let $(\overrightarrow{\mathbf{P}_k}, \mathbf{G})$ be an instance of p-HOM$(\overrightarrow{\mathcal{P}})$. The reduction produces the instance $(\mathbf{G}', s, t, k+2)$ where $\mathbf{G}'$ has vertices $G' := \{s, t\} \cup ([k] \times G)$ and as edges the symmetric closure of

$$\{((i,u),(i+1,v)) \mid i \in [k-1], (u,v) \in E^{\mathbf{G}}\}$$
$$\cup \left(\{s\} \times ([1] \times G)\right) \cup \left(\{t\} \times ([k] \times G)\right).$$

p-st-PATH $\leq_{\mathrm{pl}} p$-HOM$(\overrightarrow{\mathcal{C}}_{\mathrm{odd}})$. Let $(\mathbf{G}, s, t, k)$ be an instance of the former problem; by the previous reduction, we may assume that it is a yes instance if and only if there is an s-t path of length exactly k. We can assume that k is odd (otherwise we take a new neighbor of the given s as our new s). Define the graph $\mathbf{G}'$ with vertices $([k] \times G)$ and edges as follows. When $i \in [k-1]$ and $(u,v) \in E^G$, there is an edge from (i,u) to $(i+1,v)$; also, there is an edge from (k,t) to $(1,s)$. Then $(\mathbf{G}, s, t, k) \mapsto (\overrightarrow{\mathbf{C}_k}, \mathbf{G}')$ is a reduction as desired.

Finally, we show the hardness of p-HOM$(\mathcal{C}_{\mathrm{odd}})$. By appeal to Lemma 3.9, it suffices to demonstrate a reduction p-st-PATH $\leq_{\mathrm{pl}} p$-HOM$(\mathcal{C}_{\mathrm{odd}}^*)$. Given an instance $(\mathbf{G}, s, t, k)$ of the former problem of the above form, we define $\mathbf{G}'$ as in the previous reduction. The produced instance is $(\mathbf{C}_k^*, \mathbf{G}'')$, where $\mathbf{G}''$ is the expansion of the symmetric closure of $\mathbf{G}'$ with $C_i^{\mathbf{G}''} = \{i\} \times G$. $\square$

5 The class TREE

We give a machine characterization of the class of parameterized problems that are pl-reducible to p-HOM$(\mathcal{T}^*)$.

Definition 5.1. The class TREE contains a parameterized problem (Q, κ) if there are a computable function $f : \mathbb{N} \to \mathbb{N}$ and an alternating Turing machine that accepts Q, is pl-space bounded with respect to κ, and uses $f(\kappa(x)) \cdot \log|x|$ nondeterministic bits and $f(\kappa(x))$ co-nondeterministic bits.

The following proposition is straightforward to verify.

Proposition 5.2. *The complexity class* TREE *is closed under pl-reductions.*

Definition 5.3. An *alternating Turing machine with jumps* is a Turing machine $\mathbb{A}$ using nondeterministic jumps and a universal guess state (see Preliminaries). It *accepts* an input $x \in \{0,1\}^*$ if its starting configuration on x is *accepting*: it is already explained what an accepting halting configuration is, and a non-halting configuration which is not in the universal guess state (resp. is in the universal guess state) is accepting if at least one (resp. both) of its successor configurations are accepting.

Lemma 5.4. *Let (Q, κ) be a parameterized problem. The following are equivalent.*

1. *$(Q, \kappa) \in$ TREE*

2. *There exists a computable $f : \mathbb{N} \to \mathbb{N}$ and an alternating Turing machine $\mathbb{A}$ with $f \circ \kappa$ many jumps and $f \circ \kappa$ many co-nondeterministic bits that accepts Q and is pl-space bounded with respect to κ.*

Proof. The implication from (1) to (2) can be seen analoguously to the corresponding implication in Lemma 4.5.

Conversely, let $\mathbb{A}$ and f accord (2). A machine $\mathbb{B}$ according (1) can be obtained by simulating a jump of $\mathbb{A}$ by existentially guessing a binary string encoding a number $m \in [n]$ and moving the input head to cell m. $\qquad\square$

Theorem 5.5. $p\text{-HOM}(\mathcal{T}^*)$ *is complete for* TREE *under pl-reductions.*

Proof. (Theorem 5.5) We show that $p\text{-HOM}(\mathcal{T}^*) \in$ TREE. Consider the following alternating Turing machine. Given an instance $(\mathbf{T}, \mathbf{B})$ of $p\text{-HOM}(\mathcal{T}^*)$, the machine chooses some $t \in T$ as a "root" and computes the directed "tree" $\mathbf{T}'$ with edges directed away from t. It existentially guesses ($O(\log |B|)$ bits encoding) a $b \in C_t^B$ and writes (t, b) on some tape. While the pair (t, b) written on the tape is such that t has children in $\mathbf{T}'$ the machine does the following: universally guess ($O(\log |T|)$ bits encoding) a child t' of t; existentially guess $b' \in B$; check that $(b, b') \in E^B$ and $b' \in C_{t'}^B$. The while loop is left rejecting if this check fails. If the machine leaves the while loop otherwise, it accepts.

The number of universal guesses is bounded by $O(|T| \cdot \log |T|)$. The number of existential guesses is bounded by $|T| \cdot \log |B|$. The machine uses space to store $\mathbf{T}'$ and at most two pairs in $T \times B$, so it is pl-space bounded.

To show $p\text{-HOM}(\mathcal{T}^*)$ is TREE-hard under pl-reductions, let the parameterized problem (Q, κ) be in TREE. Choose an alternating machine $\mathbb{A}$ with jumps according to Lemma 5.4 for (Q, κ). By adding some dummy jumps and dummy universal guesses we can assume that $\mathbb{A}$ on every x and every run on x first makes one universal guess, then one jump, then one universal guess and so on. We can further assume that $\mathbb{A}$ on x on every run on x makes exactly $f(\kappa(x))$ many jumps and exactly $f(\kappa(x))$ many universal guesses. Let $\mathbb{A}^0$ ($\mathbb{A}^1$) be the machine obtained from $\mathbb{A}$ by fixing the transition from a configuration with universal guess state to the first (second) successor configuration. Note $\mathbb{A}^0$ and $\mathbb{A}^1$ are Turing machines with jumps.

Let $x \in \{0,1\}^*, k := \kappa(x), n := |x|$. Recall the proof of Theorem 4.3. As there, let $c_1, \dots, c_m$ enumerate all configurations of $\mathbb{A}$ on x with the starting state; assume c_1 is the starting configuration. Let $\mathbb{A}_{det}^0$ and $\mathbb{A}_{det}^1$ be the deterministic machines obtained form $\mathbb{A}^0$ and $\mathbb{A}^1$ by interpreting the jump state as a rejecting halting state. For

24

$i, j \in [m], b \in \{0, 1\}$ we define what it means that i *b-reaches* j as in the proof of Theorem 4.3 with $\mathbb{A}_{det}^b$ in place of $\mathbb{A}_{det}$ there; call i *accepting* if $\mathbb{A}_{det}^0$ (equivalently $\mathbb{A}_{det}^1$) started on c_i accepts in at most m steps without entering the universal guess state.

Recall the notation $\mathbf{T}_k$ from the preliminaries. The reduction outputs $(\mathbf{T}_{f(k)+1}^*, \mathbf{B})$ where $\mathbf{B}$ is defined as follows.

$$B := \{0, 1\}^{\leq f(k)+1} \times [m],$$

$E^{\mathbf{B}} :=$ the symmetric closure of
$$\left\{ ((\sigma, j), (\sigma b, j')) \mid b \in \{0, 1\}, \sigma \in \{0, 1\}^{\leq f(k)}, j \ b\text{-reaches } j' \right\},$$

$C_\lambda^{\mathbf{B}} := \{(\lambda, 1)\}, \quad \text{where } \lambda \text{ is the empty string,}$

$C_\sigma^{\mathbf{B}} := \{\sigma\} \times [m], \quad \text{for } 1 \leq |\sigma| \leq f(k),$

$C_\sigma^{\mathbf{B}} := \{(\sigma, j) \mid j \text{ is accepting}\}, \quad \text{for } |\sigma| = f(k) + 1.$

It is not hard to see that $(\mathbf{T}_{f(k)+1}^*, \mathbf{B})$ can be computed in pl-space (cf. Proof of Theorem 4.3). To see this indeed defines a reduction, first assume h is a homomorphism from $\mathbf{T}_{f(k)+1}^*$ to $\mathbf{B}$. As h preserves the unary relations C_σ, for every σ there is an $i_\sigma \in [m]$ such that $h(\sigma) = (\sigma, i_\sigma)$. It follows by induction on ℓ that for every $\sigma \in \{0, 1\}^{f(k)+1-\ell}$ the configuration c_{i_σ} is accepting (Definition 5.3). But $i_\lambda = 1$, so $c_{i_\lambda} = c_1$ is the starting configuration and $\mathbb{A}$ accepts x.

Conversely, assume $\mathbb{A}$ accepts x. We define an accepting configuration c_σ for every $\sigma \in \{0, 1\}^{\leq f(k)+1}$: c_λ is the starting configuration c_1. All other c_σs are going to be the result of a jump (are a successor of a configuration in the jump state). Assume c_σ is already defined. Then c_σ is the starting configuration or results from a jump. In both cases the machine $\mathbb{A}$ reaches from c_σ deterministically a universal guess state with two accepting successors c_0', c_1'. For every $b \in \{0, 1\}$, $\mathbb{A}$ reaches deterministically from c_b' either an accepting halting configuration or a configuration in the jump state. In the first case let $c_{\sigma b}$ be this accepting halting configuration and in the second let it be some accepting successor of the jump. For every σ choose $i_\sigma \in [m]$ such that $c_\sigma = c_{i_\sigma}$. Then $\sigma \mapsto (\sigma, i_\sigma)$ defines a homomorphism from $\mathbf{T}_{f(k)+1}^*$ to $\mathbf{B}$. $\qquad\square$

Theorem 5.6. *Let $\mathcal{A}$ be a decidable class of structures of bounded arity and bounded treewidth. Then $p\text{-}\mathrm{EMB}(\mathcal{A}) \in \mathrm{TREE}$.*

Proof. Let $\mathcal{A}$ accord the assumption. We proceed as in the proof of Theorem 3.13.

Claim. There exists a decidable class of connected structures $\mathcal{A}'$ of bounded treewidth such that $p\text{-}\mathrm{EMB}(\mathcal{A}) \leq_{\mathrm{pl}} p\text{-}\mathrm{EMB}(\mathcal{A}')$.

Note $p\text{-}\mathrm{EMB}(\mathcal{A}') \leq_{\mathrm{pl}} p\text{-}\mathrm{HOM}((\mathcal{A}')^*)$ by Lemma 3.15, the latter problem pl-reduces to $p\text{-}\mathrm{HOM}(\mathcal{T}^*)$ by the Classification Theorem, and $p\text{-}\mathrm{HOM}(\mathcal{T}^*) \in \mathrm{TREE}$ by Theorem 5.5. We are thus left to prove the claim.

Assume $\mathcal{A}$ has treewidth at most w. Fix a computable function that maps every $\mathbf{A} \in \mathcal{A}$ to a width $\leq w + 1$ tree decomposition $(\mathbf{T}, (X_t)_{t \in T})$ of $\mathbf{A}$ such that $|X_t| \geq 2$ for all $t \in T$, and $X_s \cap X_t \neq \emptyset$ for all $(s, t) \in E^{\mathbf{T}}$. Let $\mathbf{A}'$ be the expansion of $\mathbf{A}$ by interpreting a new binary relation symbol R by $\bigcup_{t \in T} X_t^2$. Then $(\mathbf{T}, (X_t)_{t \in T})$ is also a tree decomposition of $\mathbf{A}'$ and $\mathbf{A}'$ is connected. Clearly, $\mathcal{A}' := \{\mathbf{A}' \mid \mathbf{A} \in \mathcal{A}\}$ is

decidable. The map $(\mathbf{A}, \mathbf{B}) \mapsto (\mathbf{A}', \mathbf{B}')$, where $\mathbf{B}'$ is the expansion of $\mathbf{B}$ interpreting R by B^2, is a pl-reduction from $p\text{-}\textsc{Emb}(\mathcal{A})$ to $p\text{-}\textsc{Emb}(\mathcal{A}')$. $\qquad\square$

Theorem 5.7. *The parameterized problems $p\text{-}\textsc{Hom}(\mathcal{B})$, $p\text{-}\textsc{Hom}(\overrightarrow{\mathcal{B}})$, $p\text{-}\textsc{Emb}(\mathcal{B})$, and $p\text{-}\textsc{Emb}(\overrightarrow{\mathcal{B}})$ are complete for* TREE *under pl-reductions.*

Proof. It is straightforward to verify that the structures in $\mathcal{B}$ and in $\overrightarrow{\mathcal{B}}$ are connected cores. Hence, each of the first two problems is TREE-complete by the Classification Theorem and Theorem 5.5.

The problems $p\text{-}\textsc{Emb}(\mathcal{B})$ and $p\text{-}\textsc{Emb}(\overrightarrow{\mathcal{B}})$ are TREE-hard by Corollary 3.16 and the hardness of $p\text{-}\textsc{Hom}(\mathcal{B})$ and $p\text{-}\textsc{Hom}(\overrightarrow{\mathcal{B}})$, which immediately imply the hardness of $p\text{-}\textsc{Hom}(\mathcal{B}^*)$ and $p\text{-}\textsc{Hom}(\overrightarrow{\mathcal{B}}^*)$.

The problems $p\text{-}\textsc{Emb}(\mathcal{B})$ and $p\text{-}\textsc{Emb}(\overrightarrow{\mathcal{B}})$ are in TREE by Theorem 5.6. $\qquad\square$

6 Counting classification

In this section we present a classification of the counting problems corresponding to the problems $p\text{-}\textsc{Hom}(\mathcal{A})$.

6.1 Preliminaries on parameterized counting complexity

A machine with oracle $O \subseteq \{0,1\}^*$ has an extra write-only *oracle tape*; such a machine has a *query state* and the word y written on the oracle tape is the *query* of a configuration with this state; the successor state is obtained by erasing the oracle tape and moving to one of two distinguished states depending of whether the query is contained in the oracle O or not. The oracle tape is not accounted for in space bounds (as in [23]).

A *parameterized counting problem* is a pair (F, κ) of a function $F : \{0,1\}^* \to \mathbb{N}$ and a parameterization κ. To say it is in para-L, means that F is implicitly pl-computable with respect to κ. Equivalently one could say that there is a Turing machine with a write-only output tape that computes F and is pl-space bounded with respect to κ.

A *parsimonius fpt-reduction* from (F, κ) to another parameterized counting problem (F', κ') is a function $R : \{0,1\}^* \to \{0,1\}^*$ that is computable by an fpt-time bounded (with respect to κ) Turing machine such that $F = F' \circ R$ and $\kappa' \circ R \le f \circ \kappa$ for some computable $f : \mathbb{N} \to \mathbb{N}$. In the logspace setting we define a *parsimonious pl-reduction* similarly demanding that the reduction is implicitly pl-computable instead of computable by a fpt-time bounded machine. We again write $(F, \kappa) \le_{\mathrm{pl}} (F', \kappa')$ if such a reduction exists.

We say (F, κ) is *pl-Turing reducible* to (F', κ') and write $(F, \kappa) \le_{\mathrm{pl}}^{T} (F', \kappa')$ if there are a pl-space bounded (with respect to κ) Turing machine $\mathbb{A}$ with oracle to $\textsc{Bitgraph}(F')$ that decides $\textsc{Bitgraph}(F)$, and a computable f such that on every input $x \in \{0,1\}^*$ all queries $y \overset{?}{\in} \textsc{Bitgraph}(F')$ of $\mathbb{A}$ on x have parameter $\kappa'(y) \le f(\kappa(x))$. Here, we denote the parameterizations of $\textsc{Bitgraph}(F)$ and $\textsc{Bitgraph}(F')$ again by κ and κ' respectively.

6.2 Classification theorem

For a class of structures $\mathcal{A}$ consider the parameterized counting problem.

$p\text{-}\#\textsc{Hom}(\mathcal{A})$	
Instance:	A pair of structures $(\mathbf{A}, \mathbf{B})$ where $\mathbf{A} \in \mathcal{A}$.
Parameter:	$|\mathbf{A}|$.
Problem:	Compute the number of homomorphisms from $\mathbf{A}$ to $\mathbf{B}$.

Dalmau and Jonsson [10] gave a classification of counting problems of this form, showing that for a class of structures $\mathcal{A}$ of bounded arity, the problem $p\text{-}\#\textsc{Hom}(\mathcal{A})$ is in FPT if $\mathcal{A}$ has bounded treewidth, and is #W[1]-complete otherwise. We give a fine classification of the case where $\mathcal{A}$ has bounded treewidth, analogous to our fine classification for the problem $p\text{-}\textsc{Hom}(\mathcal{A})$.

Theorem 6.1 (Counting Classification)**.** *Let $\mathcal{A}$ be a decidable class of structures having bounded arity and bounded treewidth.*

1. *If $\mathcal{A}$ has unbounded pathwidth, then*

$$p\text{-}\#\textsc{Hom}(\mathcal{A}) \leq_{\mathrm{pl}} p\text{-}\#\textsc{Hom}(\mathcal{T}^*) \leq_{\mathrm{pl}}^{T} p\text{-}\#\textsc{Hom}(\mathcal{A}).$$

2. *If $\mathcal{A}$ has bounded pathwidth and unbounded tree depth, then*

$$p\text{-}\#\textsc{Hom}(\mathcal{A}) \leq_{\mathrm{pl}} p\text{-}\#\textsc{Hom}(\mathcal{P}^*) \leq_{\mathrm{pl}}^{T} p\text{-}\#\textsc{Hom}(\mathcal{A}).$$

3. *If $\mathcal{A}$ has bounded tree depth, then*

$$p\text{-}\#\textsc{Hom}(\mathcal{A}) \in \text{para-L}.$$

The proof of this result partly involves an analysis of the proof of Theorem 3.1, and builds on techniques of Dalmau and Jonsson [10].

Lemma 6.2. *Let $\mathcal{A}$ be a decidable set of finite structures, let $\mathcal{G}$ be the set of Gaifman graphs of $\mathcal{A}$, and let $\mathcal{M}$ be the set of minors of graphs in $\mathcal{G}$. Then*

$$p\text{-}\#\textsc{Hom}(\mathcal{M}^*) \leq_{\mathrm{pl}} p\text{-}\#\textsc{Hom}(\mathcal{G}^*) \leq_{\mathrm{pl}} p\text{-}\#\textsc{Hom}(\mathcal{A}^*) \leq_{\mathrm{pl}}^{T} p\text{-}\#\textsc{Hom}(\mathcal{A}).$$

Proof. The first two reductions are exactly as before, that is, they are the reductions from Lemmas 3.7 and 3.8. These reductions are readily verified to be parsimonious. We thus prove that $p\text{-}\#\textsc{Hom}(\mathcal{A}^*) \leq_{\mathrm{pl}}^{T} p\text{-}\#\textsc{Hom}(\mathcal{A})$.

Let $\mathbf{A}$ be an element of $\mathcal{A}$, and let $(\mathbf{A}^*, \mathbf{B})$ be an instance of $p\text{-}\#\textsc{Hom}(\mathcal{A}^*)$. Let $\mathbf{B}_0$ be the restriction of $\mathbf{B}$ to relation symbols of $\mathbf{A}$. For each non-empty subset $S \subseteq A$, define $\mathbf{B}_S$ to be the induced substructure of $\mathbf{A} \times \mathbf{B}_0$ on universe $\{(a, b) \in S \times B \mid b \in C_a^{\mathbf{B}}\}$. For a mapping g from A to a set of the form B_S, let g_1 denote the map $(\pi_1 \circ g)$ where π_1 is the projection of a pair to its first component.

The number of homomorphisms g from $\mathbf{A}^*$ to $\mathbf{B}$ is the same as the number M_g of homomorphisms g' from $\mathbf{A}$ to $\mathbf{B}_A$ such that g_1' is the identity on A (consider the bijection $g \mapsto g'$ with $g'(a) := (a, g(a))$). To compute M_g, it suffices to compute

the number M_h of homomorphisms h from $\mathbf{A}$ to $\mathbf{B}_A$ such that $h_1(A) = A$. This is because of the fact that a mapping $h : A \to A \times B$ is a homomorphism from $\mathbf{A}$ to $\mathbf{B}_A$ with $h_1(A) = A$ if and only if h has the form $g \circ \sigma$ where g is a homomorphism from $\mathbf{A}$ to $\mathbf{B}_A$, g_1 is the identity, and σ is a bijective homomorphism from $\mathbf{A}$ to $\mathbf{A}$. From this fact, it follows that $M_g = M_h/S$ where S is the number of bijective homomorphisms from $\mathbf{A}$ to $\mathbf{A}$; note that S can be computed directly from $\mathbf{A}$, and so this gives a way to determine M_g (division is logspace computable [7]). We prove the claimed fact as follows. The backward direction is clear, so we prove the forward direction. Let h be a homomorphism from $\mathbf{A}$ to $\mathbf{B}_A$ with $(\pi_1 \circ h)(A) = A$. There exists an integer $m \geq 1$ such that $(\pi_1 \circ h)^m$ is the identity mapping on A. Set $g = h \circ (\pi_1 \circ h)^{m-1}$; we then have $h = h \circ (\pi_1 \circ h)^m = g \circ (\pi_1 \circ h)$, as desired.

For each subset $S \subseteq A$, the Turing reduction will query the instance $(\mathbf{A}, \mathbf{B}_S)$ of p-#HOM($\mathcal{A}$); denote the result by $N_{\subseteq S}$. Observe that $N_{\subseteq S}$ is the number of homomorphisms h from $\mathbf{A}$ to $\mathbf{B}_A$ with $h_1(A) \subseteq S$. For a subset $S \subseteq A$, let $N_{=S}$ denote the number of homomorphisms h from $\mathbf{A}$ to $\mathbf{B}_A$ with $h_1(A) = S$. We have, for each subset $S \subseteq A$, the identity $N_{\subseteq S} = \sum_{T \subseteq S} N_{=T}$. By inclusion-exclusion, we have $N_{=A} = \sum_{S \subseteq A} (-1)^{|A|-|S|} N_{\subseteq S}$ which is the value M_h that we wanted to determine. We can evaluate the sum expression in pl-space by combining two observations: first, with oracle access to p-#HOM($\mathcal{A}$) the sequence of the numbers $(-1)^{|A|-|S|} N_{\subseteq S}$ is implicitly pl-computable; second, summing a sequence of integers can be done in logspace. $\qquad\qquad\square$

Proof of Theorem 6.1. Statements (1) and (2) each make two claims. The claims made first concern parsimonious pl-reductions and follow from Lemma 3.4 and Remark 3.5. The second claims concern Turing reductions and follow from the previous lemma together with the Excluded Tree Theorem 2.3 (2) and the Excluded Path Theorem 2.3 (3) respectively.

We are left to prove Statement (3). It is not hard to see that a structure of tree depth at most w' has a tree decomposition of width at most $w' + 1$ such that the underlying tree has height at most w' with respect to some root. By Lemma 3.4 and Remark 3.5, it suffices to show p-#HOM($\mathcal{T}(w)$) $\in$ para-L for every $w \in \mathbb{N}$. Here, we let $\mathcal{T}(w)$ be the class of structures $\mathbf{T}^*$ such that $\mathbf{T}$ is a tree that can be rooted in such a way that its height is at most w.

For $w = 0$, this is easy to see. So let $w > 0$ and assume by induction that p-#HOM($\mathcal{T}(w-1)$) $\in$ para-L. Given an instance $(\mathbf{T}^*, \mathbf{B})$ of p-#HOM($\mathcal{T}(w)$), we conceive of $\mathbf{T}^*$ as a rooted tree with root r and of height at most w. For elements $t \in T$ and $b \in B$, we define $N_{t \to b}$ to be the number of partial homomorphisms h that are defined on the subtree rooted at t and such that $h(t) = b$. Let $t_1, \ldots, t_m$ denote the children of r in $\mathbf{T}$. The number that we desire to determine is $\sum_{b \in C_r^{\mathbf{B}}} N_{r \to b}$. For a particular value $b \in B$, it is straightforward to verify that

$$N_{r \to b} = \prod_{i=1}^m \sum_{b'} N_{t_i \to b'}$$

where the sum is over all $b' \in C_{t_i}^{\mathbf{B}}$ such that $(b, b') \in E^{\mathbf{B}}$. Thus, the number we desire to compute equals a certain sum-product-sum expression. But this expression is implicitly pl-computable: to determine bits of the numbers $N_{t_i \to b'}$ one can run an

algorithm witnessing $p\text{-}\#\textsc{Hom}(\mathcal{T}(w-1)) \in$ para-L. Using the facts that iterated sum and iterated product are computable in logarithmic space [7], it follows that our sum-product-sum expression can be evaluated in logarithmic space. This yields the result. □

7 Discussion

A classification of the parameterized complexity of embedding problems is famously open [17, p.355], in particular, it is not known whether the embedding problem for complete bipartite cliques is W[1]-hard (under fpt-reductions).

A fundamental problem whose complexity we failed to settle within our framework is $p\text{-}\textsc{Emb}(\mathcal{P})$. By Theorem 4.6, we know $p\text{-}\textsc{Emb}(\mathcal{P}) \in$ PATH, but we do not know whether it is PATH-hard (under pl-reductions). We note that its restriction to regular graphs is in para-L.

$p\text{-}\textsc{Emb}(\mathcal{P})_{\text{reg}}$	
Instance:	A regular graph $\mathbf{G}$ and $k \in \mathbb{N}$.
Parameter:	k.
Problem:	Does $\mathbf{G}$ contain a path of length k?

Proposition 7.1. $p\text{-}\textsc{Emb}(\mathcal{P})_{\text{reg}} \in$ para-L.

The proof uses a result of Flum and Grohe [15, Example 6] stating that model checking first-order logic on bounded degree graphs is in para-L. Their proof actually shows

Theorem 7.2 ([15]). *The following parameterized problem is in* para-L.

Instance:	A graph $\mathbf{G}$ of degree at most d and a first-order sentence φ.
Parameter:	$d + |\varphi|$.
Problem:	$\mathbf{G} \models \varphi$?

Proof of Proposition 7.1. Given a regular graph $\mathbf{G}$ of degree d and a natural $k \in \mathbb{N}$, distinguish two cases: if $d > k$ then accept; otherwise check, using the algorithm of Theorem 7.2, whether $\mathbf{G}$ satisfies $\exists x_0 \cdots x_k \big(\bigwedge_{i<j\leq k} \neg x_i = x_j \wedge \bigwedge_{i<k} E x_i x_{i+1} \big)$. □

On the more structural side, as mentioned in the introduction, we believe that it could be worthwhile to investigate whether or not the classes PATH and TREE are closed under complement. Relatedly, one can ask whether or not it holds that co-PATH $\subseteq$ TREE.

Acknowledgements

References

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[2] A. Blumensath and B. Courcelle. On the monadic second-order transduction hierarchy. *Logical Methods in Computer Science*, 6(2), 2010.

[3] H. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209:1–45, 1998.

[4] A. K. Chandra and P. M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *Proceddings of STOC'77*, pages 77–90, 1977.

[5] H. Chen and M. Grohe. Constraint satisfaction with succinctly specified relations. *Journal of Computer and System Sciences*, 76(8):847–860, 2010.

[6] Y. Chen, J. Flum, and M. Grohe. Bounded nondeterminism and alternation in parameterized complexity theory. *18th IEEE Conference on Computational Complexity*, pages 13–29, 2003.

[7] A. Chiu, G. Davida, and B. Litow. Division in logspace-uniform NC^1. *Theoretical Informatics and Applications*, 35:259–275, 2001.

[8] N. Creignou, S. Khanna, and M. Sudan. *Complexity Classification of Boolean Constraint Satisfaction Problems*. SIAM Monographs on Discrete Mathematics and Applications. Society for Industrial and Applied Mathematics, 2001.

[9] N. Creignou, P. G. Kolaitis, and H. Vollmer, editors. *Complexity of Constraints - An Overview of Current Research Themes*, volume 5250 of *Lecture Notes in Computer Science*. Springer, 2008.

[10] V. Dalmau and P. Jonsson. The complexity of counting homomorphisms seen from the other side. *Theoretical Computer Science*, 329:315–323, 2004.

[11] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.

[12] M. Elberfeld, C. Stockhusen, and T. Tantau. On the space complexity of parameterized problems. *7th International Symposium of Parameterized and Exact Computation, Springer LNCS*, 7535:206–217, 2012.

[13] T. Feder and M. Y. Vardi. The computational structure of monotone monadic snp and constraint satisfaction: A study through datalog and group theory. *SIAM Journal on Computing*, 28(1):57–104, 1999.

[14] J. Flum, M. Frick, and Grohe. Query evaluation via tree-decompositions. *Journal of the ACM*, 49(6):716–752, 2002.

[15] J. Flum and M. Grohe. Describing parameterized complexity classes. *Information and Computation*, 187:291–319, 2003.

[16] J. Flum and M. Grohe. The parameterized complexity of counting problems. *SIAM Journal on Computing*, 33:892–922, 2006.

[17] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.

[18] G. Gottlob, N. Leone, and F. Scarcello. The complexity of acyclic conjunctive queries. *Journal of the ACM*, 48(3):431–498, 2001.

[19] G. Gottlob, N. Leone, and F. Scarcello. Hypertree decompositions and tractable queries. *J. Comput. Syst. Sci.*, 64(3):579–627, 2002.

[20] M. Grohe. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *Journal of the ACM*, 54(1), 2007.

[21] M. Grohe, T. Schwentick, and L. Segoufin. When is the evaluation of conjunctive queries tractable? In *STOC 2001*, 2001.

[22] P. Kolaitis and M. Y. Vardi. Conjunctive-query containment and constraint satisfaction. *17th ACM Symposium on Principles of Database Systems*, 10:205–213, 1998, full version at: `http://www.cs.rice.edu/~vardi/papers`.

[23] R. E. Ladner and N. A. Lynch. Relativization of questions about log space computability. *Mathematical Systems Theory*, 10:19–32, 1976.

[24] D. Marx. Can you beat treewidth? *Theory of Computing*, 6(1):85–112, 2010.

[25] D. Marx. Tractable hypergraph properties for constraint satisfaction and conjunctive queries. In *Proceedings of the 42nd ACM Symposium on Theory of Computing*, pages 735–744, 2010.

[26] J. Nešetřil and P. O. de Mendez. Tree depth, subgraph coloring, and homomorphism bounds. *European Journal of Combinatorics*, 27(6):1022–1041, 2006.

[27] C. Papadimitriou and M. Yannakakis. On the Complexity of Database Queries. *Journal of Computer and System Sciences*, 58(3):407–427, 1999.

[28] N. Schweikardt, T. Schwentick, and L. Segoufin. Database theory: Query languages. In M. J. Atallah and M. Blanton, editors, *Algorithms and Theory of Computation Handbook*, volume 2: Special Topics and Techniques, chapter 19. CRC Press, second edition, Nov 2009.