A Dichotomy for Succinct Representations of Homomorphisms

Christoph Berkholz
Technische Universität Ilmenau, Germany

Harry Vinall-Smeeth
Humboldt-Universität zu Berlin, Germany

Abstract

The task of computing homomorphisms between two finite relational structures $\mathcal{A}$ and $\mathcal{B}$ is a well-studied question with numerous applications. Since the set $\mathrm{Hom}(\mathcal{A}, \mathcal{B})$ of all homomorphisms may be very large having a method of representing it in a succinct way, especially one which enables us to perform efficient enumeration and counting, could be extremely useful.

One simple yet powerful way of doing so is to decompose $\mathrm{Hom}(\mathcal{A}, \mathcal{B})$ using union and Cartesian product. Such data structures, called d-representations, have been introduced by Olteanu and Závodný [32] in the context of database theory. Their results also imply that if the treewidth of the left-hand side structure $\mathcal{A}$ is bounded, then a d-representation of polynomial size can be found in polynomial time. We show that for structures of bounded arity this is optimal: if the treewidth is unbounded then there are instances where the size of any d-representation is superpolynomial. Along the way we develop tools for proving lower bounds on the size of d-representations, in particular we define a notion of reduction suitable for this context and prove an almost tight lower bound on the size of d-representations of all k-cliques in a graph.

2012 ACM Subject Classification Theory of computation → Data structures and algorithms for data management; Theory of computation → Complexity theory and logic; Theory of computation → Data compression

Keywords and phrases homomorphism problem, CSP, succinct representations, enumeration, lower bound, treewidth

Digital Object Identifier 10.4230/LIPIcs.ICALP.2023.113

Category Track B: Automata, Logic, Semantics, and Theory of Programming

Related Version *Full Version*: `https://arxiv.org/abs/2209.14662`

Funding *Christoph Berkholz*: Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 414325841.
Harry Vinall-Smeeth: Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 385256563.

1 Introduction

The task of computing homomorphisms between two finite relational structures has a long history and numerous applications. Most notably, as pointed out by Feder and Vardi [17], it is the right abstraction for the constraint satisfaction problem (CSP) – a framework for search problems that generalised Boolean satisfiability. Moreover, evaluating conjunctive queries on a relational database is equivalent to computing homomorphisms from the query structure to the database. While deciding the existence of a homomorphism from a structure $\mathcal{A}$ to a structure $\mathcal{B}$ is a classical NP-complete problem, several restrictions of the input instance have been considered in order to understand the landscape of tractability. One line of research investigates *right-hand-side* restrictions, where it is asked for which classes of structures $\mathcal{B}$ the CSP becomes tractable and when it remains hard. This culminated in

50th International Colloquium on Automata, Languages, and Programming (ICALP 2023).
Editors: Kousha Etessami, Uriel Feige, and Gabriele Puppis; Article No. 113; pp. 113:1–113:19
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

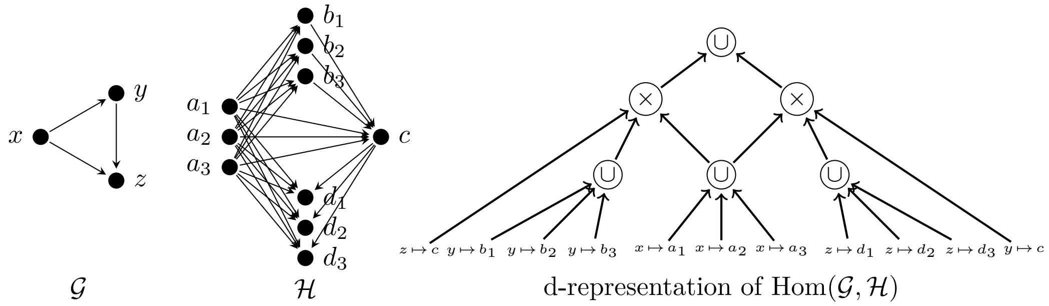

Figure 1 A deterministic d-representation of all homomorphisms from $\mathcal{G}$ to $\mathcal{H}$.

the solution [9, 38] of the CSP-dichotomy conjecture [17] that characterises those $\mathcal{B}$ where finding a homomorphism from a given structure $\mathcal{A}$ can be done in polynomial time (assuming P≠NP).

Another line of research, to which we contribute in this paper, focuses on *left-hand-side* restrictions: for which classes of structures $\mathcal{A}$ can we efficiently find a homomorphism from $\mathcal{A}$ to a given $\mathcal{B}$? In this scenario, a dichotomy is only known when all relations have bounded arity, as is the case for graphs, digraphs, or k-uniform hypergraphs. Grohe [20] showed that, modulo complexity theoretic assumptions, for any class of structures $\mathfrak{A}$ of bounded arity the decision problem, "Given a structure $\mathcal{A} \in \mathfrak{A}$ and a structure $\mathcal{B}$, is there a homomorphism from $\mathcal{A}$ to $\mathcal{B}$?" is in polynomial time if and only if the homomorphic core of every structure in $\mathfrak{A}$ has bounded treewidth. For classes of unbounded arity, polynomial time tractability has been shown for fractional hypertreewidth [5, 21], but a full characterisation of tractability has only been obtained in the parameterised setting using submodular width [29]. Besides deciding the existence of a homomorphism, the complexity of counting all homomorphism has also been characterised in the right-hand-side regime [8] and for bounded-arity classes of left-hand-side structures [14]. A third task, that is less well understood, is to *enumerate* all homomorphisms; here only partial results on the complexity are known (e. g. [10, 13, 37, 19]).

In this work we consider the task of *representing* the set $\mathrm{Hom}(\mathcal{A}, \mathcal{B})$ of all homomorphisms in a succinct and accessible way. In particular, we want to store all, potentially exponentially many, homomorphisms, in a data structure of polynomial size that enables us to, e. g. generate a stream of all homomorphisms. The data structures we are interested in – so-called d-representations – were first introduced to represent homomorphisms in the context of join evaluation under the name *factorised databases* [32]. They are conceptually very simple: the set of homomorphisms is represented by a circuit, where the "inputs" are mappings of single vertices and larger sets of mappings are generated by combining local mappings using Cartesian product × and union ∪. In the circuit previously computed sets of local homomorphisms are represented by gates and can be used several times, see Figure 1 for an example. Such a representation is called *deterministic* if every ∪-gate is guaranteed to combine disjoint sets. Deterministic representations have the advantage that the number of homomorphisms can be efficiently counted by adding the sizes of the local homomorphism sets on every ∪-gate and multiplying them on every ×-gate. Moreover, all homomorphisms can be efficiently enumerated where the delay between two outputs is only linear in the size of every produced homomorphism (= size of the universe of $\mathcal{A}$) [2]. It is known that if the treewidth of the left-hand side structure is bounded, then a deterministic d-representation of polynomial size can be found in polynomial time [32]. Our main theorem shows that for structures of bounded arity this is optimal: if the treewidth is unbounded, then there are instances where the size of any (not necessarily deterministic) representation is superpolynomial.

▶ **Theorem 1.** *Let $r \in \mathbb{N}$, σ a signature of arity $\leq r$ and $\mathfrak{A}$ a class of σ-structures. Then the following are equivalent:*

1. *There is a $w \in \mathbb{N}$ such that every structure in $\mathfrak{A}$ has treewidth at most w.*
2. *A deterministic d-representation of $\mathrm{Hom}(\mathcal{A}, \mathcal{B})$ can be computed in polynomial time, for any $\mathcal{A} \in \mathfrak{A}$ and any $\mathcal{B}$.*
3. *There is a $c \in \mathbb{N}$ such that for any $\mathcal{A} \in \mathfrak{A}$ and any $\mathcal{B}$ there exists a (not necessarily deterministic) d-representation of $\mathrm{Hom}(\mathcal{A}, \mathcal{B})$ of size $O\big((\|\mathcal{A}\| + \|\mathcal{B}\|)^c\big)$.*

Related work. The research on succinct data structures for homomorphism problems has emerged from the two different perspectives. When fixing the right-hand-side structure $\mathcal{B}$, then data structures like multi-valued decision diagrams (MDD) [4], AND/OR multi-valued decision diagrams (AOMDD) [30], and multi-valued decomposable decision graphs (MDDG) [25] have been proposed, which arose from representations for Boolean functions that are studied in *knowledge compilation* (see, e. g., [15]). The (deterministic) d-representations studied in this paper can be interpreted as (deterministic) DNNF circuits with zero-suppressed semantics [2, Lemma 7.4], where a $\cup$-gate corresponds to a (deterministic) $\vee$-gate and a $\times$-gate corresponds to a *decomposable* $\wedge$-gate.

In the left-hand-side regime, representations have been introduced in the context of enumerating query results. Most notably, Olteanu and Závodnỳ [32] introduced the notion of *factorised databases* that are used to decompose the result relation of a conjunctive query using Cartesian product and union. Their findings imply the upper bound part of our dichotomy theorem: if $\mathcal{A}$ has bounded treewidth, its tree decomposition defines a so-called *d-tree*, which structures the polynomial size d-representation. They have also shown a limited lower bound for *structured* representations ("d-representations respecting a d-tree"). However, this lower bound considers only a small subclass of all possible d-representations. In a similar vein, in knowledge compilation there exist several restrictions of DNNFs e. g. requiring $\vee$-gates to be decision or deterministic [15], or enforcing structuredness [33]. In this light, the significance of our lower bound comes from the fact that it holds for the most general notion of representations (d-representations), which correspond to unrestricted DNNFs.

The proof of our lower bound has some connections to the conditional lower bound for the counting complexity of homomorphisms [14], which in turn builds upon the construction of Grohe [20]. The essence of these proofs is to rely on an assumption about the hardness of the parametrised clique problem and reduce this to all structures of unbounded treewidth. We take a similar route: in Section 5 we prove an unconditional lower bound for representing cliques and obtain our main lower bound using a sequence of reductions in Section 6.

The circuit notion for representing the set of homomorphisms between two given structures (or, equivalently, the result relation of a multiway join query) in a succinct data structure might be confused with previous work on the (Boolean or arithmetic) circuit complexity for deciding or counting homomorphisms or subgraphs. In this research branch, a structure $\mathcal{B}$ over a universe of size n is given as input to a circuit $C_{\mathcal{A},n}$, which decides the existence of or counts the number of homomorphisms (or subgraph-embeddings) from $\mathcal{A}$ to $\mathcal{B}$. Examples include monotone circuits for finding cliques [1, 36], bounded-depth circuits for finding cliques and other small subgraphs [35, 26] as well as graph polynomials and monotone arithmetic circuits [16, 7, 24] for counting homomorphisms. In particular, the recent work of Komarath, Pandey and Rahul [24] studies monotone arithmetic circuits that have, for each pattern $\mathcal{G}$ and each n, an input indicator variable $x_{\{u,v\}}$ for each potential edge $\{u, v\} \in [n]^2$ in the second graph $\mathcal{H}$. For every input (i.e. setting indicator variables according to a graph $\mathcal{H}$

on n vertices), the arithmetic circuit has to compute the number of homomorphisms from $\mathcal{G}$ to $\mathcal{H}$. Interestingly, Komorath et al. prove a tight bound and show that such arithmetic circuits need size $n^{\mathrm{tw}(\mathcal{G})+1}$. Unfortunately, this and related results from circuit complexity (such as lower bounds for the clique problem) do not translate to the knowledge compilation approach. Part of the reason is that we crucially have a different representation *for each pair* $\mathcal{G}$, $\mathcal{H}$ and having, e. g. an arithmetic circuit computing the constant number $|\mathrm{Hom}(\mathcal{G},\mathcal{H})|$ is trivial. Moreover, due to monotonicity, the worst-case right-hand-side instances $\mathcal{H}$ in [24] are complete graphs, whereas d-representations lack this form of monotonicity: adding edges to $\mathcal{H}$ can make factorisation simpler and in particular occurences of patterns in complete graphs can be succinctly factorised.

Despite this, some techniques on a more general level (e. g. arguing about the transversal of a circuit or using random graphs as bad examples) are useful in circuit complexity as well as for proving lower bounds on representations.

2 Preliminaries

We write $\mathbb{N}$ for the set of non-negative integers and define $[n] := \{1, \dots, n\}$ for any positive integer n. Given a set S we write 2^S to denote the power set of S. Whenever writing a to denote a k-tuple, we write a_i to denote the tuple's i-th component; i. e., $a = (a_1, \dots, a_k)$. For a function $f \colon X \to Y$ and $X' \subset X$ we write $\pi_{X'} f$ to denote the projection of f to X'. Given a set of functions, each of which has a domain containing X', we write $\pi_{X'} F := \{\pi_{X'} f \mid f \in F\}$.

Graphs, Minors, Structures, Tree Decompositions. Whenever $\mathcal{G}$ is a graph or a hypergraph we write $V(\mathcal{G})$ and $E(\mathcal{G})$ for the set of nodes and the set of edges, respectively, of $\mathcal{G}$. We let $\mathcal{K}_k$ be the complete graph on k vertices, $\mathcal{C}_k$ the k-cycle graph, and $\mathcal{G}_k$ the $k \times k$-grid graph. Given a graph $\mathcal{G}$ and $\{u, v\} \in E(\mathcal{G})$, we can form a new graph by *edge contraction*: replacing u and v be a new vertex w adjacent to all neighbours of u and v. A graph $\mathcal{H}$ is a *minor* of $\mathcal{G}$ if $\mathcal{H}$ can be obtained from $\mathcal{G}$ by repeatedly deleting vertices, deleting edges and contracting edges.

A *tree decomposition* of a graph $\mathcal{G}$ is a pair (T, β) where T is a tree and $\beta \colon V(T) \to 2^{V(\mathcal{G})}$ associates to every node $t \in V(T)$ a bag $\beta(t)$ such that the following is satisfied: (1) For every $v \in V(\mathcal{G})$ the set $\{t \in V(T) \mid v \in \beta(t)\}$ is non-empty and forms a connected set in T. (2) For every $\{u, v\} \in E(\mathcal{G})$ there is some $t \in V(T)$ such that $\{u, v\} \subseteq \beta(t)$. The width of a tree decomposition is $\max_{t \in V(T)} |\beta(t)| - 1$ and the *treewidth of* $\mathcal{G}$ is the minimum width of any tree decomposition of $\mathcal{G}$.

A (relational) signature σ is a set of relation symbols R, each of which is equipped with an arity $r = r(R)$. A (finite, relational) σ-structure $\mathcal{A}$ consists of a finite universe A and relations $R^{\mathcal{A}} \subseteq A^r$ for every r-ary relation symbol $R \in \sigma$. We will write $\|\mathcal{A}\| := \sum_{R \in \sigma} |R^{\mathcal{A}}|$. The *Gaifman graph* of $\mathcal{A}$ is the graph with vertex set A and edges $\{u, v\}$ for any distinct u, v that occur together in a tuple of a relation in $\mathcal{A}$. The *treewidth* of a structure is the treewidth of its Gaifman graph. We say $\mathcal{A}$ is *connected* if its Gaifman graph is connected and we will henceforth assume, without loss of generality, that all structures in this paper are connected.

Enumeration. An enumeration algorithm for $\mathrm{Hom}(\mathcal{A}, \mathcal{B})$ proceeds in two stages. In the *preprocessing* stage the algorithm does some preprocessing on $\mathcal{A}$ and $\mathcal{B}$. In the *enumeration phase* the algorithm enumerates, without repetition, all homomorphisms in $\mathrm{Hom}(\mathcal{A}, \mathcal{B})$, followed by the end of enumeration message. The *delay* is the maximum of three times: the time between the start of the enumeration phase and the first output homomorphism, the maximum time between the output of two consecutive homomorphisms and between the last tuple and the end of enumeration message. The *preprocessing time* is the time the algorithm spends in the preprocessing stage, which may be 0. Similarly given a d-representation C for $\mathrm{Hom}(\mathcal{A}, \mathcal{B})$, an enumeration algorithm for C has a preprocessing stage, where it can do some preprocessing on C, and an enumeration phase defined as above.

3 Homomorphisms and the complexity of constraint satisfaction

A *homomorphism* $h\colon \mathcal{A} \to \mathcal{B}$ between two σ-structures $\mathcal{A}$ and $\mathcal{B}$ is a mapping from A to B that preserves all relations, i. e., for every r-ary $R \in \sigma$ and $(a_1, \ldots, a_r) \in A^r$ it holds that if $(a_1, \ldots, a_r) \in R^{\mathcal{A}}$, then $(h(a_1), \ldots, h(a_r)) \in R^{\mathcal{B}}$. We let $\mathrm{Hom}(\mathcal{A}, \mathcal{B})$ be the set of all homomorphisms from $\mathcal{A}$ to $\mathcal{B}$. A (homomorphic) core of a structure $\mathcal{A}$ is an inclusion-wise minimal substructure $\mathcal{A}' \subseteq \mathcal{A}$ such that there is a homomorphism from $\mathcal{A}$ to $\mathcal{A}'$. It is well known that all cores of a structure are isomorphic, hence we will also speak of *the* core of a structure.

Following common notation we fix a (potentially infinite) signature σ and define for classes of σ-structures $\mathfrak{A}$ and $\mathfrak{B}$ the (promise) decision problem $\mathrm{CSP}(\mathfrak{A}, \mathfrak{B})$ to be: "Given two σ-structures $\mathcal{A} \in \mathfrak{A}$ and $\mathcal{B} \in \mathfrak{B}$, is there a homomorphism from $\mathcal{A}$ to $\mathcal{B}$?" Similarly, the counting problem $\#\mathrm{CSP}(\mathfrak{A}, \mathfrak{B})$ asks: "Given two σ-structures $\mathcal{A} \in \mathfrak{A}$ and $\mathcal{B} \in \mathfrak{B}$, what is the number of homomorphisms from $\mathcal{A}$ to $\mathcal{B}$?" A lot of work has been devoted towards classifying the classes of structures for which the problems are solvable in polynomial time. Normally either the left-hand-side $\mathfrak{A}$ or the right-hand-side $\mathfrak{B}$ is restricted and the other part ($\mathfrak{B}$ or $\mathfrak{A}$) is the class $\_$ of all structures. A related problem is $\mathrm{Enum\text{-}CSP}(\mathfrak{A}, \mathfrak{B})$ [10], which is the following task: "Given two σ-structures $\mathcal{A} \in \mathfrak{A}$ and $\mathcal{B} \in \mathfrak{B}$, enumerate all homomorphisms from $\mathcal{A}$ to $\mathcal{B}$". One way of defining tractability for enumeration algorithms is *polynomial delay enumeration*, where the preprocessing time and the delay is polynomial in $\mathcal{A}$ and $\mathcal{B}$.

In this paper we focus on "left-hand-side" restrictions, where $\mathfrak{B}$ is the class of all structures. Moreover, we assume that the arity of each symbol in σ is bounded by some constant r. In this setting the complexity of $\mathrm{CSP}(\mathfrak{A}, \_)$ and $\#\mathrm{CSP}(\mathfrak{A}, \_)$ is fairly well understood: the decision problem is polynomial time tractable iff the core of every structure in $\mathfrak{A}$ has bounded treewidth, while the counting problem is tractable if every structure from $\mathfrak{A}$ itself has bounded treewidth. This is made precise by the following two theorems.

▶ **Theorem 2** ([20]). *Let $r \in \mathbb{N}$, σ be a signature of arity $\leq r$ and $\mathfrak{A}$ a class of σ-structures. Under the assumption that there is no $c \in \mathbb{N}$ such that for every $k \in \mathbb{N}$ there is an algorithm that finds a k-clique in an n-vertex graph in time $O(n^c)$ the following two statements are equivalent.*

1. *There is a $w \in \mathbb{N}$ such that the core of every structure in $\mathfrak{A}$ has treewidth at most w.*
2. $\mathrm{CSP}(\mathfrak{A},\_)$ *is solvable in polynomial time.*

▶ **Theorem 3** ([14]). *Let $r \in \mathbb{N}$, σ be a signature of arity $\leq r$ and $\mathfrak{A}$ a class of σ-structures. Under the assumption that there is no $c \in \mathbb{N}$ such that for every $k \in \mathbb{N}$ there is an algorithm that counts the number of k-cliques in an n-vertex graph in time $O(n^c)$ the following two statements are equivalent.*

1. *There is a $w \in \mathbb{N}$ such that every structure in $\mathfrak{A}$ has treewidth at most w.*
2. $\#\mathrm{CSP}(\mathfrak{A},\_)$ *is solvable in polynomial time.*

To understand the difference between these characterisations, consider the class $\mathfrak{A}$ of all structures $\mathcal{A}_k$ that are complete graphs on k vertices with an additional vertex with a self-loop. The homomorphic core of such structures is just the self-loop and finding one homomorphism from $\mathcal{A}_k$ to $\mathcal{B}$ is equivalent to finding a self-loop in $\mathcal{B}$. However, counting homomorphisms from $\mathcal{A}_k$ to $\mathcal{B}$ is as hard as counting k-cliques: if $\mathcal{B}$ is a simple graph $\mathcal{G}$ with one additional vertex with a self-loop, then the number of homomorphisms from $\mathcal{A}_k$ to $\mathcal{B}$ is the number of k-cliques in $\mathcal{G}$ plus one.

The complexity of the corresponding enumeration problem Enum-CSP($\mathfrak{A}$, __) is still open. It has been shown that polynomial delay enumeration is possible if $\mathfrak{A}$ has bounded treewidth [10]. On the other hand, polynomial delay enumeration implies solvability of the decision problem in polynomial time (because either the first solution or an end-of-enumeration message has to appear after polynomial time). Hence it follows from Theorem 2, under the same complexity assumption, that there is no polynomial delay enumeration algorithm if the cores of the structures in $\mathfrak{A}$ have unbounded treewidth. For further discussion on this topic we refer the reader to [10].

Our main result (Theorem 1) can be viewed as an unconditional dichotomy for enumeration and counting in a restricted class of algorithms: when the algorithm relies on local decompositions into union and product, then the tractable instances are exactly those that have bounded treewidth. Interestingly, this matches the conditional dichotomy for the counting case (Theorem 3).

4 Factorised Representations

In this section we formally introduce the factorisation formats for CSPs. These formats agree with the factorised representations of relations introduced by Olteanu and Závodný [32] in the context of evaluating conjunctive queries on relational databases. While we stick to the naming conventions introduced there we provide a slightly different circuit-based definition that is very much inspired by [2] and the notion of *set circuits* introduced in [3].

A *factorisation circuit* C for two sets A and B is an acyclic directed graph with node labels and a unique sink. Each node without incoming edges is called an *input gate* and labelled by $\{a \mapsto b\}$ for some $a \in A$ and $b \in B$. Every other node is labelled by either $\cup$ or $\times$ and called a $\cup$-*gate* or $\times$-*gate*, respectively. For each gate g in the circuit we inductively define its *domain* $\mathrm{dom}(g) \subseteq A$ by $\mathrm{dom}(g) = \{a\}$ if g is an input gate with label $\{a \mapsto b\}$ and $\mathrm{dom}(g) = \bigcup_{i=1}^{r} \mathrm{dom}(g_i)$ if g is a non-input gate with child gates $g_1, \dots, g_r$.

A factorisation circuit is *well-defined* if for every gate g with child gates $g_1, \dots, g_r$ it holds that $\mathrm{dom}(g) = \mathrm{dom}(g_1) = \cdots = \mathrm{dom}(g_r)$ if g is a $\cup$-gate and $\mathrm{dom}(g_i) \cap \mathrm{dom}(g_j) = \emptyset$ for all $i \neq j$ if g is a $\times$-gate. For every gate g in a well-defined factorisation circuit we let S_g be a set of mappings $h\colon \mathrm{dom}(g) \to B$ defined by

$$S_g := \begin{cases} \{\{a \mapsto b\}\} & \text{if } g \text{ is an input labelled by } \{a \mapsto b\} \\ S_{g_1} \cup \cdots \cup S_{g_r} & \text{if } g \text{ is a } \cup\text{-gate with children } g_1, \dots, g_r, \\ \{\, h_1 \cup \cdots \cup h_r \mid h_i \in S_{g_i},\ i \in [r] \,\} & \text{if } g \text{ is a } \times\text{-gate with children } g_1, \dots, g_r. \end{cases} \tag{1}$$

We define $S_C := S_s$ for the sink s of C. For each gate g we let C_g be the sub-circuit with sink g. By $\|C\|$ we denote the size of a factorisation circuit C, which is defined to be the number of gates plus the number of wires. The number of gates in C is denoted by $|C|$.

Before defining factorised representations for CSP-instances, we introduce two special types of circuits. A factorisation circuit is *treelike* if the underlying graph is a tree, i. e., every non-sink gate has exactly one parent. Moreover, a well-defined factorisation circuit is *deterministic* if for every $\cup$-gate g the set S_g is a *disjoint* union of its child sets $S_{g_1}, \dots, S_{g_r}$. Note that while treelikeness is a syntactic property of the circuit structure, being deterministic is a semantic property that depends on the valuations of the gates. Now we are ready to state a circuit-based definition of the factorised representations defined in [32].

▶ **Definition 4.** *Let $\mathcal{A}$ and $\mathcal{B}$ be two σ-structures.*

1. *A (deterministic)* d-representation *for $\mathcal{A}$ and $\mathcal{B}$ is a well-defined (deterministic) factorisation circuit over $V(\mathcal{A})$ and $V(\mathcal{B})$ where $S_C = \mathrm{Hom}(\mathcal{A}, \mathcal{B})$.*
2. *A (deterministic)* f-representation *is a (deterministic) d-representation with the additional restriction that the circuit is treelike.*

For brevity we will sometimes refer to d/f-representations as d/f-reps. Note that a d-rep can be more succinct than a f-rep and we will mostly deal with d-reps in this paper. However, in the proofs it will sometimes be convenient to expand out the circuit in order to make it treelike. More formally, the *transversal* $\mathrm{Trans}(C)$ of a d-rep C is the f-rep obtained from C as follows: using a top-down transversal starting at the output gate, we replace each gate g with parents $p_1, \dots, p_d$ by d copies $g_1, \dots, g_d$ such that the in-edges of each g_i are exactly the children of g and g_i has exactly one out-edge going to p_i. This procedure produces a treelike circuit that is well-defined/deterministic if C was well-defined/deterministic. Finally it can easily be verified that $S_{\mathrm{Trans}(C)} = S_C$.

We will often want to construct new factorised circuits from old ones. The following lemma introduces two constructions that will be particularly useful, the proof of correctness can be found in the full version of this paper.

▶ **Lemma 5.** *Let $\mathcal{A}, \mathcal{B}$ be σ-structures and C be a d-rep of $\mathrm{Hom}(\mathcal{A}, \mathcal{B})$. Let $X = \{x_1, \dots, x_\ell\} \subseteq A$, $Y_1, \dots, Y_\ell \subseteq B$, $\ell \geq 1$. Then one can construct the following factorised circuits in time $O(\|C\|)$.*

1. *C', such that $S_{C'} = \pi_X \mathrm{Hom}(\mathcal{A}, \mathcal{B})$ and $\|C'\| \leq \|C\|$.*
2. *C'', such that $S_{C''} = \{h \in \mathrm{Hom}(\mathcal{A}, \mathcal{B}) \mid h(x_i) \in Y_i, i \in [\ell]\}$ and $\|C''\| \leq \|C\|$.*

A special f-rep is the *flat representation*: a depth-2 circuit with a single ∪-gate at the top followed by a layer of ×-gates. Note that for any pair $\mathcal{A}$, $\mathcal{B}$, of σ-structures the flat representation has size $1 + |\mathrm{Hom}(\mathcal{A}, \mathcal{B})| \cdot (2|A| + 2)$. Intuitively, this representation corresponds to listing all homomorphisms and provides a trivial upper bound on representation size.

Deterministic d-representations have two desirable properties: they allow us to compute $|\mathrm{Hom}(\mathcal{A}, \mathcal{B})|$ in time $O(\|C\|)$ and to enumerate all homomorphisms with $O(|A|)$ delay after $O(\|C\|)$ preprocessing. Efficient counting is possible by computing bottom-up the number $|S_g|$ for each gate using multiplication on every ×-gate and summation on every (deterministic) ∪-gate. If, additionally, C is normal – i. e., no parent of a ∪-gate is a ∪-gate and the in-degree of every ∪- and ×-gate is at least 2 – Olteanu and Závodný [32, Theorem 4.11] show enumeration with $O(|A|)$ delay and *no* preprocessing is possible by sequentially enumerating the sets S_{g_i} of every child of a (deterministic) ∪-gate and by a nested loop to generate all combinations of child elements at ×-gates. The case where C is not normal is shown in [2, Theorem 7.5] and is more involved. Note that the delay is optimal in the sense that every homomorphism that is enumerated is of size $O(|A|)$.

In the other direction this means that constructing a deterministic d-rep is at least as hard as counting the number of homomorphisms. Our main theorem implies that, modulo the same assumptions as Theorem 3, the opposite is also true: for a class $\mathfrak{A}$ of structures of bounded arity there is a polynomial time algorithm that constructs a d-representation of polynomial size for two given structures $\mathcal{A} \in \mathfrak{A}$ and $\mathcal{B}$ if and only if, there is a polynomial-time algorithm that counts the number of homomorphisms between $\mathcal{A} \in \mathfrak{A}$ and $\mathcal{B}$.

Upper bounds on representation size. We have already argued that there is always a flat representation of size $O(|A| \cdot |\mathrm{Hom}(\mathcal{A}, \mathcal{B})|)$. Thus, as a corollary of [5] we get an upper bound of $O(|A| \cdot \|\mathcal{B}\|^{\rho^*(\mathcal{A})})$, where $\rho^*(\mathcal{A})$ is the *fractional edge cover number of $\mathcal{A}$*. Note that, however, the fractional edge cover number for structures of bounded arity is quite large. More precisely, if all relations in $\mathcal{A}$ have arity at most r, then $\rho^*(\mathcal{A}) \geq \frac{1}{r}|A|$.

Luckily in many cases we can do better: the results by Olteanu and Závodný in [32] imply that given a tree-decomposition of $\mathcal{A}$ of width $w-1$ we can construct a d-rep of $\mathrm{Hom}(\mathcal{A},\mathcal{B})$ of size $O(\|\mathcal{A}\|^2\|\mathcal{B}\|^w)$ in time $O(\mathrm{poly}(\|\mathcal{A}\|)\|\mathcal{B}\|^w\log(\|\mathcal{B}\|))$. Moreover the d-reps produced are normal and deterministic, meaning they allow us to perform efficient enumeration and counting. Therefore if $\mathfrak{A}$ is a class of bounded treewidth this gives us one method for solving $\#\mathrm{CSP}(\mathfrak{A},\_)$ in polynomial time. In fact, the same holds true if w is the more general *fractional hypertreewidth*, although for the case of bounded arity structures the two measure differ only by a constant. We discuss the unbounded arity case in the conclusion and, in more detail, in the full version of this paper.

5 A near-optimal bound for cliques

The goal of this section is to prove the following two theorems:

▶ **Theorem 6.** *For any $k \in \mathbb{N}$ there exist arbitrary large graphs $\mathcal{G}$ with m edges such that any f-rep of $\mathrm{Hom}(\mathcal{K}_k,\mathcal{G})$ has size $\Omega(m^{k/2}/\log^k(m))$.*

▶ **Theorem 7.** *For any $k \in \mathbb{N}$ there exist arbitrary large graphs $\mathcal{G}$ with m edges such that any d-rep of $\mathrm{Hom}(\mathcal{K}_k,\mathcal{G})$ has size $\Omega(m^{k/2}/\log^{3k-1}(m))$.*

These bounds are almost tight since the number of k-cliques in a graph with m edges is bounded by $m^{k/2}$. Moreover Theorem 7 is a crucial ingredient for proving our main theorem in Section 6. We will first prove Theorem 6 and then show how this implies the bound for d-reps.

The main idea is to exploit a correspondence between the structure of a (simple) graph $\mathcal{G}$ and f-reps of $\mathrm{Hom}(\mathcal{K}_k,\mathcal{G})$. To illustrate this consider the case $k=2$, where $V(\mathcal{K}_2)=\{x_1,x_2\}$ and each $h \in \mathrm{Hom}(\mathcal{K}_2,\mathcal{G})$ corresponds to an edge of $\mathcal{G}$. Let C be a f-rep of $\mathrm{Hom}(\mathcal{K}_2,\mathcal{G})$, with ×-gates $g_1,\dots,g_\alpha$. Each g_i has two children g_i^1, g_i^2 with $\mathrm{dom}(g_i^1)=x_1$ and $\mathrm{dom}(g_i^2)=x_2$. Since no ×-gates can occur in $C_{g_i^1}$ or $C_{g_i^2}$, $S^1_{g_i}=\{\{x_1\mapsto a\}\mid a\in A_i\}$ and $S_{g_i^2}=\{\{x_2\mapsto b\}\mid b\in B_i\}$ for some disjoint $A_i,B_i\subseteq V(\mathcal{G})$. Therefore $A_i\times B_i$ is a complete bipartite subgraph of $\mathcal{G}$. Since the ancestors of each ×-gate can only be ∪-gates, each f-rep of $\mathrm{Hom}(\mathcal{K}_2,\mathcal{G})$ corresponds to a set of complete bipartite subgraphs that cover every edge of $\mathcal{G}$. Finding such sets and investigating their properties has been studied in various contexts, for example see [12, 18, 22, 31].

Moreover, the number of input gates appearing in C is $\sum_{i=1}^{\alpha}|A_i|+|B_i|$ and so finding a f-rep of $\mathrm{Hom}(\mathcal{K}_2,\mathcal{G})$ of minimum size corresponds to minimising the sum of the sizes of the partitions in our complete bipartite covering of $\mathcal{G}$, call this the cost of the covering. Proving Theorem 6 for the case $k=2$, corresponds to finding graphs where every covering of the edges by complete bipartite subgraphs has high cost. This is a problem investigated by Chung et al. in [12], where one key idea is that if a graph contains no large complete bipartite subgraphs and a large number of edges then the cost of any cover must be high. We deploy this idea in our more general context. This motivates the following lemma, which follows from a simple probabilistic argument.

▶ **Lemma 8.** *For every $k \in \mathbb{N}$ there exists some $c_k \in \mathbb{R}^+$ such that for every sufficiently large integer n there is a graph $\mathcal{G}$ with n vertices, such that*

1. *$\mathcal{G}$ has $m \geq \frac{1}{8}n^2$ edges,*
2. *$\mathcal{G}$ contains no complete bipartite subgraph $\mathcal{K}_{a,a}$ for $a \geq 3\log(n)$, and*
3. *the number of k-cliques in $\mathcal{G}$ is at least $c_k n^k$.*

Proof. We first prove the following claim.

▷ Claim 9. Let $\mathcal{G}_n$ be a random graph on n vertices with edge probability $\frac{1}{2}$. Let $\epsilon > 0$. Then for any $a = a(n) \geq (2+\epsilon)\log(n)$,

$$P_a := \mathbb{P}(\mathcal{G}_n \text{ has } \mathcal{K}_{a,a} \text{ as a subgraph}) \to 0 \text{ as } n \to \infty.$$

Proof of Claim. By the union bound and the bound on a we get

$$P_a \leq \binom{n}{a}^2 2^{-a^2} \leq n^{2a} 2^{-a^2} = 2^{2a\log(n)-a^2} \leq 2^{-(\epsilon^2+2\epsilon)\log^2 n}. \qquad \triangleleft$$

Now let $\mathcal{G}_n$ be as above, $s = s(k) := \binom{k}{2} + 1$ and p be the probability that such a graph has at least $\binom{n}{k}2^{-s}$ k-cliques. The expected number of k-cliques in $\mathcal{G}_n$ is $\binom{n}{k}2^{-\binom{k}{2}}$. Therefore,

$$\binom{n}{k}2^{-\binom{k}{2}} \leq \binom{n}{k}2^{-s}(1-p) + \binom{n}{k}p$$

and so $p \geq 1/(2^s - 1)$. Moreover, by the Chernoff bound, (1) from the statement of the Lemma fails only with exponentially small probability. By Claim 9 there must exist a $\mathcal{G}$ satisfying (1), (2), and (3) for sufficiently large n. ◀

Equipped with Lemma 8 we are already in a position to prove Theorem 6.

Proof of Theorem 6. Let $\mathcal{G}$ be an n-vertex graph provided by Lemma 8 and suppose that C is a f-rep for $\mathcal{K}_k$ and $\mathcal{G}$. If $\max_{x\in \text{dom}(g)} |\{a \mid h(x) = a, h \in S_g\}| \leq 3\log(n)$ for a gate g we say that g is *small*. Otherwise we say g is *big*. Note that a ×-gate cannot have two big children g_1 and g_2 because otherwise there would be $x_1 \in \text{dom}(g_1)$ and $x_2 \in \text{dom}(g_2)$ such that

$$\{a \mid h(x_1) = a, h \in S_{g_1}\} \times \{a \mid h(x_2) = a, h \in S_{g_2}\}$$

forms a complete bipartite subgraph with partitions bigger than $3\log n$ in $\mathcal{G}$, contradicting (2) from Lemma 8.

If g is small, then C_g represents $|S_g| \leq 3^{|\text{dom}(g)|}\log^{|\text{dom}(g)|}(n)$ homomorphisms. We claim that for any gate g of C, $|S_g| \leq |C_g| \cdot 3^{|\text{dom}(g)|}\log^{|\text{dom}(g)|}(n)$. Clearly this holds for input gates. We can therefore induct bottom up on C. Suppose our claim holds for all children $g_1, \ldots, g_r$ of some gate g. If g is a ×-gate then we know at most one of the g_i is big, say g_1. Define $b := \sum_{i=2}^{r} |\text{dom}(g_i)|$. Then,

$$|S_g| = \prod_{i=1}^{r} |S_{g_i}| \leq |C_{g_1}| \cdot 3^{|\text{dom}(g_1)|}\log^{|\text{dom}(g_1)|}(n) \cdot 3^b \log^b(n) \leq |C_g| \cdot 3^{|\text{dom}(g)|}\log^{|\text{dom}(g)|}(n),$$

The ∪-gate case follows immediately from the induction hypothesis because the circuit is treelike so if g has children $g_1, \ldots, g_r$ then $|C_g| = 1 + \sum_{i=1}^{r} |C_{g_i}|$.

From the claim we infer in particular that $|\text{Hom}(\mathcal{K}_k, \mathcal{G})| = |S_s| \leq |C| \cdot 3^k \log^k(n)$ for the sink s of C. By (3) from Lemma 8 it follows that $|C| \geq c_k n^k / (3^k \log^k(n))$ which, combined with (1) from Lemma 8, implies the claimed result. ◀

We now transfer this bound to d-reps, by showing that, for the same graphs used above, any d-rep cannot be much smaller than the smallest f-rep.

Proof of Theorem 7. Let $\mathcal{G}$ be an n-vertex graph provided by Lemma 8 as above and C a d-rep of $\text{Hom}(\mathcal{K}_k, \mathcal{G})$ with sink s. If a gate has out-degree of more than one we call it a *definition*. As in the proof of Theorem 6, if $\max_{x\in \text{dom}(g)} |\{a \mid h(x) = a, h \in S_g\}| \leq 3\log(n)$ for a gate g we say that g is *small*. Otherwise we say g is *big*.

Our strategy is to convert C into an equivalent f-rep that is not much bigger than C. For ease of analysis and exposition we will do this by first eliminating all small definitions and then all big definitions. First if s is small replace the whole circuit with its equivalent flat representation. Otherwise, we mark all small gates g that have a big parent and compute the equivalent flat representation F_g of C_g. Since every unmarked small gate is a descendant of some marked gate, we can now safely delete all unmarked small gates. Afterwards we consider every wire between a marked gate g and one of its big parents p and replace it by a copy of F_g as input to p. We obtain an equivalent circuit $\hat{C}$ where every small gate has only one parent. The size (number of gates plus number of wires) increases only by a factor determined by the maximum size of a flat representation:

▷ Claim 10. $\|\hat{C}\| \leq \|C\| \cdot (2k+3)3^k \log^k(n)$.

When we try and eliminate big definitions one challenge is that if g is big, then $\|\hat{C}_g\|$ can be large and so making lots of copies of it could blow up the size of our circuit. To overcome this we introduce the notion of an *active parent*. We then show that non-active parents are effectively redundant and that there can't be too many active ones, which allows us to construct an equivalent treelike circuit of the appropriate size.

So let g be a definition with parents $p_1, \ldots, p_\alpha$, $\alpha > 1$, and suppose there is a unique path from p_i to the sink s for every i. Then for every gate v on the unique path from g to s which passes through p_i, we inductively define a set of (partial) homomorphisms $A_i^v = A_i^v(g)$ as follows, where $\hat{v}$ refers to the child of v also lying on this path.

- $A_i^g := S_g$,
- if v is a $\cup$-gate $A_i^v := A_i^{\hat{v}}$,
- otherwise v is a $\times$-gate with children $u_1, \ldots, u_{r-1}, \hat{v}$ and $A_i^v := \{h_1 \cup \ldots \cup h_r \mid h_i \in S_{u_i}, i \in [r-1], h_r \in A_i^{\hat{v}}\}$.

Write $A_i := A_i^s$, intuitively this is the set of homomorphisms that the wire from g to p_i contributes to. We say that a parent p_i of g is *active* if $A_i \nsubseteq \cup_{j \neq i} A_j$. Now using a top-down traversal starting at the output gate of $\hat{C}$, we replace each gate with active parents $p_1, \ldots, p_\beta$, by β copies $g_1, \ldots, g_\beta$ such that the children of each g_i are exactly the children of g and g_i has exactly one out-edge going to p_i. At each stage we also clean-up the circuit by iteratively deleting all gates which have no incoming wires, as well as all the wires originating from such gates. We can think of this process as constructing a slimmed down version of the traversal, where at each stage we only keep wires going to active parents. Call the resulting circuit C'.

We first note that this process is well-defined, as there is a unique path from the sink to itself and since whenever we visit a gate we have already visited all of its parents. Moreover, by construction this results in a treelike circuit. In the next claim we bound the size of C' and show it is indeed an equivalent circuit. The idea is that firstly a gate cannot have too many active parents, as otherwise we would get a large biclique in $\mathcal{G}$ which is ruled out by Lemma 8, and secondly that since only active parents contribute new homomorphisms we really do get an equivalent circuit, see the full version of this paper for details.

▷ Claim 11. C' is a f-rep of $\mathrm{Hom}(\mathcal{K}_k, \mathcal{G})$ and $\|C'\| \leq 3^k \log^{k-1}(n) \|\hat{C}\|$.

Pulling everything together we get that

$$\|C\| \underset{\text{(Claim 10)}}{\geq} \frac{\|\hat{C}\|}{(2k+3)3^k \log^k(n)} \underset{\text{(Claim 11)}}{\geq} \frac{\|C'\|}{(2k+3)3^{2k} \log^{2k-1}(n)} = \Omega\left(\frac{m^{k/2}}{\log^{3k-1}(m)}\right),$$

where the final equality follows by Theorem 6 since C' is a f-rep of $\mathrm{Hom}(\mathcal{K}_k, \mathcal{G})$. ◀

6 The representation dichotomy for structures of bounded arity

In this section we lift the lower bound for cliques to all classes of graphs with unbounded treewidth. We first introduce a notion of reductions between representations and show that having lower bounds for all graphs of unbounded treewidth immediately implies our main dichotomy theorem for bounded-arity structures.

Afterwards, we introduce minor and almost-minor reductions and use them to obtain a lower bound for representing homomorphisms from large grids and from graphs having large grids as a minor. The superpolynomial representation lower bound for all graph classes with unbounded treewidth then follows from the excluded grid theorem.

6.1 Reductions between representations

In order to define reductions between representations we fix some notation. For two structures $\mathcal{A}$ and $\mathcal{B}$ we let $\mathsf{D}(\mathcal{A},\mathcal{B})$ be the set of all d-representations of $\mathrm{Hom}(\mathcal{A},\mathcal{B})$ and $\mathsf{d}(\mathcal{A},\mathcal{B}) = \min_{C\in\mathsf{D}(\mathcal{A},\mathcal{B})} \|C\|$ be the size of the smallest such representation.

For a class $\mathfrak{C}$ of structures the function $\mathsf{d}_{\mathcal{A},\mathfrak{C}}\colon \mathbb{N} \to \mathbb{N}$ expresses the required size of a d-representation of homomorphisms between $\mathcal{A}$ and $\mathcal{C} \in \mathfrak{C}$ in terms of the size m of $\mathcal{C}$, i. e., $\mathsf{d}_{\mathcal{A},\mathfrak{C}}(m) = \max_{\{\mathcal{C}\in\mathfrak{C}\,:\,\|\mathcal{C}\|\leq m\}} \mathsf{d}(\mathcal{A},\mathcal{C})$. We write $\mathsf{d}_{\mathcal{A}}$ as an abbreviation for $\mathsf{d}_{\mathcal{A},\mathfrak{C}}$ when $\mathfrak{C}$ is the class of all structures. Translated to this notation, [32] showed that $\mathsf{d}_{\mathcal{A}} = O(m^{\mathrm{tw}(\mathcal{A})+1})$, whereas Theorem 7 states the lower bound $\mathsf{d}_{K_k} = \Omega(m^{k/2}/\log^{3k-1}(m))$. We also write, for a signature σ, $\mathfrak{C}_\sigma$ to denote the class of all σ-structures.

The main goal of this section is to prove, for some increasing function f, a lower bound of the form $\mathsf{d}_{\mathcal{A}} = \Omega(m^{f(\mathrm{tw}(\mathcal{A}))/\,\mathsf{ar}(\mathcal{A})})$ for every structure $\mathcal{A}$, which immediately implies our main theorem. To achieve this we use reductions with our k-clique lower bound as a starting point. Suppose we already have a lower bound on $\mathsf{d}_{\mathcal{A},\mathfrak{C}}$ for a class $\mathfrak{C}$ of arbitrarily large hard instances (implying a lower bound on $\mathsf{d}_{\mathcal{A}}$), then we can use the following reduction *from* $\mathcal{A}$ *to* $\mathcal{B}$ *via* $\mathfrak{C}$ to obtain a lower bound on $\mathsf{d}_{\mathcal{B}}$.

▶ **Definition 12.** *Let $\mathcal{A}$ be a σ-structure and let $\mathfrak{C}$ be a class of σ-structures. Let $\mathcal{B}$ be a σ'-structure and $c\colon \mathbb{R}^+ \to \mathbb{R}^+$ be a strictly increasing function. Then a* c-reduction *from $\mathcal{A}$ to $\mathcal{B}$ via $\mathfrak{C}$ is a pair $(\phi,(\psi_{\mathcal{C}})_{\mathcal{C}\in\mathfrak{C}})$, where $\phi\colon \mathfrak{C} \to \mathfrak{C}_{\sigma'}$ and $\psi_{\mathcal{C}}\colon \mathsf{D}(\mathcal{B},\phi(\mathcal{C})) \to \mathsf{D}(\mathcal{A},\mathcal{C})$ such that:*

1. *for every $n \in \mathbb{N}$ there is a $\mathcal{C} \in \mathfrak{C}$ such that $\|\phi(\mathcal{C})\| \geq n$,*
2. *$\|\phi(\mathcal{C})\| \leq c(\|\mathcal{C}\|)$ for all $\mathcal{C} \in \mathfrak{C}$, and*
3. *$\|\psi_{\mathcal{C}}(C)\| \leq \|C\|$ for every structure $\mathcal{C} \in \mathfrak{C}$ and circuit $C \in \mathsf{D}(\mathcal{B},\phi(\mathcal{C}))$.*

If $c(m) = \alpha m$ for some $\alpha \in \mathbb{R}^+$, we say we have a linear reduction*.*

▶ **Lemma 13.** *Suppose there is a c-reduction $(\phi,(\psi_{\mathcal{C}})_{\mathcal{C}\in\mathfrak{C}})$ from $\mathcal{A}$ to $\mathcal{B}$ via $\mathfrak{C}$, let $\mathfrak{D} = \{\phi(\mathcal{C}) \mid \mathcal{C} \in \mathfrak{C}\}$ be the image of ϕ. Then $\mathsf{d}_{\mathcal{B},\mathfrak{D}} = \Omega(\mathsf{d}_{\mathcal{A},\mathfrak{C}} \circ \lfloor c^{-1} \rfloor)$.*

Proof. Fix $m \in \mathbb{N}$, where $m \geq \min_{\{\mathcal{C}\in\mathfrak{C}\}} \|\mathcal{C}\|$. Let $\mathcal{C} \in \mathfrak{C}$ with $\|\mathcal{C}\| \leq m$. Then $\psi_{\mathcal{C}}$ witnesses that $\mathsf{d}(\mathcal{A},\mathcal{C}) \leq \mathsf{d}(\mathcal{B},\phi(\mathcal{C}))$. Also $\|\phi(\mathcal{C})\| \leq c(m)$, since c is an increasing function. So $\mathsf{d}_{\mathcal{B},\mathfrak{D}}(c(m)) = \max_{\{\mathcal{C}\,:\,\|\phi(\mathcal{C})\|\leq c(m)\}} \mathsf{d}(\mathcal{B},\phi(\mathcal{C})) \geq \max_{\{\mathcal{C}\,:\,\|\mathcal{C}\|\leq m\}} \mathsf{d}(\mathcal{A},\mathcal{C}) = \mathsf{d}_{\mathcal{A},\mathfrak{C}}(m)$. Since $\mathfrak{C}$ and $\mathfrak{D}$ contain arbitrarily large structures, the asymptotic bound from the lemma follows. ◀

We start illustrating the power of these reductions by making two simplifications. First, we reduce the general problem of representing homomorphisms to representing homomorphisms that respect a partition. Second, we further reduce to graph homomorphisms that respect a partition. All proofs from this subsection can be found in the full version of the paper.

For the first reduction we need the notion of the *individualisation* of a σ-structure $\mathcal{A}$, which is obtained from $\mathcal{A}$ by giving every element of the universe a distinct color. More precisely, we extend the vocabulary σ with unary relations (= colours) $\sigma_A = \{P_a \colon a \in A\}$ and let $\mathcal{A}^{\mathrm{id}}$ be the $\sigma \cup \sigma_A$-expansion of $\mathcal{A}$ by adding $P_a^{\mathcal{A}^{\mathrm{id}}} = \{a\}$.

▶ **Lemma 14.** *Let $\mathcal{A}$ be a σ-structure and let $\mathfrak{C}$ be the class of all $\sigma \cup \sigma_A$-structures where $\{P_a^{\mathcal{C}} \mid a \in A\}$ is a partition of the universe. Then $\mathsf{d}_{\mathcal{A}} = \Omega(\mathsf{d}_{\mathcal{A}^{id},\mathfrak{C}})$.*

We call structures and (vertex-coloured) graphs *individualised* if every vertex has a distinct colour. In the next lemma we reduce from individualised structures to individualised graphs. Recall the definition of the Gaifman graph $\mathcal{G}_{\mathcal{A}}$ from the preliminaries.

▶ **Lemma 15.** *Let $\mathcal{A}$ be an individualised structure and $\mathcal{G}_{\mathcal{A}}^{id}$ the individualisation of its Gaifman graph. Let $\mathfrak{C}$ be the class of all structures $\mathcal{C}$ where $\{P_a^{\mathcal{C}} \mid a \in A\}$ is a partition of its universe and $\mathfrak{H}$ be the class of all vertex-coloured graphs $\mathcal{H}$ where $\{P_a^{\mathcal{H}} \mid a \in A\}$ is a partition of its vertex set. Then $\mathsf{d}_{\mathcal{A},\mathfrak{C}}(m) = \Omega\big((\mathsf{d}_{\mathcal{G}_{\mathcal{A}}^{id},\mathfrak{H}}(m))^{2/\,\mathsf{ar}(\mathcal{A})} \big)$.*

Taking both lemmas into account, we can now focus on individualised graphs $\mathcal{G}$ on the left-hand side and on graphs $\mathcal{H}$ with the corresponding colouring $\{P_a^{\mathcal{H}} \mid a \in V(\mathcal{G})\}$ that partitions its vertex set on the right-hand side. We call such graphs *$V(\mathcal{G})$-partitioned graphs.* However we would also like to deploy our lower bound from Section 5; the next lemma allows to transfer this lower bound to individualised structures.

▶ **Lemma 16.** *Let $\mathcal{G}$ be a graph and $\mathfrak{C}$ be the class of all $V(\mathcal{G})$-partitioned graphs. Then $\mathsf{d}_{\mathcal{G}^{id},\mathfrak{C}} = \Omega(\mathsf{d}_{\mathcal{G}})$.*

6.2 Minor reductions

In this subsection we show that we can reduce $\mathcal{G}'$ to $\mathcal{G}$ if $\mathcal{G}$ is a minor of $\mathcal{G}'$. We start by illustrating how to handle edge contractions via an example.

▶ **Example 17** (Reduction from 4-cycle to 3-cycle)**.** Consider the 3-cycle $\mathcal{K}_3$ on vertices x_1, x_2, x_3, which is a minor of the 4-cycle $\mathcal{C}_4$ on vertices x_1, x_2, x_3, x_4 by contracting one edge $\{x_4, x_1\}$. We show that we can lift the lower bound for $\mathcal{K}_3^{\mathrm{id}}$ (Theorem 7 + Lemma 16) to $\mathcal{C}_4^{\mathrm{id}}$ (and hence $\mathcal{C}_4$ by Lemma 14) by a simple linear reduction from $\mathcal{K}_3^{\mathrm{id}}$ to $\mathcal{C}_4^{\mathrm{id}}$ via the class of all $\{x_1, x_2, x_3\}$-partitioned graphs. Let $\mathcal{H}$ be a $\{x_1, x_2, x_3\}$-partitioned graph. We define the $\{x_1, x_2, x_3, x_4\}$-partitioned graph $\mathcal{H}' = \phi(\mathcal{H})$ by $P_x^{\mathcal{H}'} := P_x^{\mathcal{H}}$ for $x \in \{x_1, x_2, x_3\}$, $P_{x_4}^{\mathcal{H}'} := \{\widehat{v} \mid v \in P_{x_1}^{\mathcal{H}}\}$ and $E(\mathcal{H}') =$

$$\begin{aligned}
&\big\{\{v, \widehat{v}\} \mid v \in P_{x_1}^{\mathcal{H}}\big\}\\
\cup\,&\big\{\{v, w\} \mid v \in P_{x_1}^{\mathcal{H}},\ w \in P_{x_2}^{\mathcal{H}},\ \{v, w\} \in E(\mathcal{H})\big\}\\
\cup\,&\big\{\{v, w\} \mid v \in P_{x_2}^{\mathcal{H}},\ w \in P_{x_3}^{\mathcal{H}},\ \{v, w\} \in E(\mathcal{H})\big\}\\
\cup\,&\big\{\{v, \widehat{w}\} \mid v \in P_{x_3}^{\mathcal{H}},\ w \in P_{x_1}^{\mathcal{H}},\ \{v, w\} \in E(\mathcal{H})\big\}.
\end{aligned}$$

Note that the size of $\mathcal{H}'$ is linear in the size of $\mathcal{H}$. The construction ensures that any mapping $h' \colon \{x_1, \dots, x_4\} \to V(\mathcal{H}')$ is a homomorphism from $\mathcal{C}_4^{\mathrm{id}}$ to $\mathcal{H}'$ if, and only if, $h'(x_4) = \widehat{h'(x_1)}$ and $h(x_i) := h'(x_i)$, for $i \in [3]$, is a homomorphism from $\mathcal{K}_3^{\mathrm{id}}$ to $\mathcal{H}$. Therefore, $\mathrm{Hom}(\mathcal{K}_3^{\mathrm{id}}, \mathcal{H}) = \pi_{\{x_1,x_2,x_3\}} \mathrm{Hom}(\mathcal{C}_4^{\mathrm{id}}, \mathcal{H}')$ and a representation C' of $\mathrm{Hom}(\mathcal{K}_3^{\mathrm{id}}, \mathcal{H})$ can be obtained from a representation C of $\mathrm{Hom}(\mathcal{C}_4^{\mathrm{id}}, \mathcal{H}')$ by Lemma 5 which, moreover, guarantees that $\|C'\| \leq \|C\|$. Therefore we do have a linear reduction from $\mathcal{C}_4^{\mathrm{id}}$ to $\mathcal{K}_3^{\mathrm{id}}$. It follows that $\mathsf{d}_{\mathcal{C}_4}(m) = \Omega(\mathsf{d}_{\mathcal{C}_4^{\mathrm{id}},\mathfrak{C}}(m)) = \Omega(\mathsf{d}_{\mathcal{K}_3^{\mathrm{id}},\mathfrak{H}}(m)) = \Omega(\mathsf{d}_{\mathcal{K}_3}(m)) = \Omega(m^{3/2}/\log^7(m))$, where $\mathfrak{C}$ is the class of $V(\mathcal{C}_4^{\mathrm{id}})$-partitioned graphs and $\mathfrak{H}$ is the class of $V(\mathcal{K}_3^{\mathrm{id}})$-partitioned graphs. The first equality follows by Lemma 14, the second by Lemma 13, the third by Lemma 16 and the last by Theorem 7.

So to handle edge contractions we take the partitioned hard right-hand side instance and "re-introduce" the edge $\{x, y\}$ contracted to x by copying P_x to P_y and adding a perfect matching between the two partitions P_x and P_y. Handling edge deletions is even simpler: suppose that $\{x, y\}$ is deleted from $\mathcal{G}'$ to $\mathcal{G}$ and we want to reduce $\mathcal{G}'$ to $\mathcal{G}$. Then we take a partitioned hard instance for $\mathcal{G}$ and just introduce the complete bipartite graph between the partitions P_x and P_y; this may square the size of the graph. Since the sets of (partition-respecting) homomorphisms are the same for both instances, we do not even have to modify the representations in the reduction. The next lemma summarises these findings. Its proof is omitted as it is subsumed by Lemma 22.

▶ **Lemma 18.** *Let $\mathcal{G}_X, \mathcal{G}_Y$ be graphs with vertex sets X and Y respectively such that $\mathcal{G}_X$ is a minor of $\mathcal{G}_Y$. Let $\mathfrak{H}$ be the class of all $V(\mathcal{G}_X)$-partitioned graphs and $\mathfrak{H}'$ the class of all $V(\mathcal{G}_Y)$-partitioned graphs. Then there is a c-reduction $(\phi, (\psi_{\mathcal{H}})_{\mathcal{H}\in\mathfrak{H}})$ from $\mathcal{G}_Y^{id}$ to $\mathcal{G}_X^{id}$ via $\mathfrak{H}$ with $\phi(\mathfrak{H}) \subseteq \mathfrak{H}'$ and $c(m) = O(m^2)$.*

This yields together with Lemmas 13, 14 and 16 along with Theorem 7 the following corollary.

▶ **Corollary 19.** *If $\mathcal{G}$ has $\mathcal{K}_k$ as a minor, then $\mathsf{d}_{\mathcal{G}} = \Omega(m^{k/4}/\log^{(3k-1)/2}(m))$.*

6.3 Relaxation of the minor condition

Every graph having $\mathcal{K}_k$ as a minor has treewidth at least $k-1$, so Corollary 19 provides the desired lower bound of Theorem 1 for certain large-treewidth graphs. However, there are graphs of large treewidth that do not have a large clique as a minor. Instead, the excluded grid theorem [34] and its more efficient version [11] tells us that graphs of large treewidth always have a large $k \times k$-grid as a minor.

▶ **Theorem 20** ([11]). *There is a polynomial function $w : \mathbb{N} \to \mathbb{N}$ such that for every k the $(k \times k)$-grid is a minor of every graph of treewidth at least $w(k)$.*

Thus, in order to prove Theorem 1 it suffices to combine Lemma 18 with a lower bound for grid graphs. We cannot reduce immediately to our k-clique lower bound, as the grid does not have a $\mathcal{K}_k$ minor for $k \geq 5$. However, the complete graph $\mathcal{K}_k$ is "almost a minor" of $\mathcal{G}_{2k-2}$ for the following notion of *almost minor* that is good enough to prove a variant of Lemma 18.

▶ **Definition 21.** *For two graphs $\mathcal{G}_X$, $\mathcal{G}_Y$ with vertex sets $X = V(\mathcal{G}_X)$ and $Y = V(\mathcal{G}_Y)$ we say that a map $M \colon Y \to 2^X$ is* almost minor *if the following conditions hold:*

1. *for every $y \in Y$, $|M(y)| \in \{1, 2\}$;*
2. *for every $x \in X$ there is a $y \in Y$ s.t. $M(y) = \{x\}$ and for every x, x' adjacent in $\mathcal{G}_X$ there exists y, y' adjacent in $\mathcal{G}_Y$ such that $M(y) = \{x\}$ and $M(y') = \{x'\}$;*
3. *for each $x \in X$, $\{y \colon x \in M(y)\}$ is connected in $\mathcal{G}_Y$ and*
4. *if $M(y) = \{x, x'\}$ with $x \neq x'$ and y' is adjacent to y in $\mathcal{G}_Y$, then $M(y') = \{x\}$ or $M(y') = \{x'\}$.*

If such a map exists we say $\mathcal{G}_X$ is an almost minor of $\mathcal{G}_Y$.

For the special case when $|M(y)| = 1$ for all y, M is a *minor map* and G_X is a minor of G_Y. The motivation for this definition is that whilst grids are planar, large cliques are not and so we introduce "junctions", i.e. nodes y such that $M(y) = \{x_1, x_2\}$ which allows

$\{v \mid x_i \in M(v)\}$, $i \in \{1,2\}$ to intersect in a controlled way, see Figure 2. We should also observe here that this notion is related to Marx's notion of an *embedding* [27].[1] Now we can state our reduction lemma for almost minors, which extends Lemma 18.

▶ **Lemma 22.** *Let $\mathcal{G}_X$, $\mathcal{G}_Y$ be the graphs with vertex sets X and Y, respectively, such that $\mathcal{G}_X$ is an almost minor of $\mathcal{G}_Y$. Let $\mathfrak{H}$ be the class of all X-partitioned graphs and $\mathfrak{H}'$ be the class of all Y-partitioned graphs, then there is a c-reduction $(\phi, (\psi_{\mathcal{H}})_{\mathcal{H}\in\mathfrak{H}})$ from $\mathcal{G}_Y^{id}$ to $\mathcal{G}_X^{id}$ via $\mathfrak{H}$ with $\phi(\mathfrak{H}) \subseteq \mathfrak{H}'$ and $c = O(m^2)$.*

Proof. We start by defining the Y-partitioned graph $\mathcal{H}^* = \phi(\mathcal{H})$ for an arbitrary X-partitioned graph $\mathcal{H}$. To define the partitions, we consider two cases: if $M(y) = \{x\}$, we let $P_y^{\mathcal{H}^*} := \{v_a^y \mid a \in P_x^{\mathcal{H}}\}$ and if $M(y) = \{x, x'\}$, then $P_y^{\mathcal{H}^*} := \{v_{\{a,b\}}^y \mid a \in P_x^{\mathcal{H}}, b \in P_{x'}^{\mathcal{H}}\}$. For every edge $\{y, y'\} \in E(\mathcal{G}_Y)$ we define the edge set $E_{\{y,y'\}}$ between the partitions $P_y^{\mathcal{H}^*}$ and $P_{y'}^{\mathcal{H}^*}$ by the following exhaustive cases:

1. if $M(y) = M(y') = \{x\}$: $E_{\{y,y'\}} := \big\{\{v_a^y, v_a^{y'}\} \mid a \in P_x^{\mathcal{H}}\big\}$
2. if $M(y) = \{x\}$, $M(y') = \{x'\}$, and $\{x, x'\} \in E(\mathcal{G}_X)$: $E_{\{y,y'\}} := \big\{\{v_a^y, v_b^{y'}\} \mid a \in P_x^{\mathcal{H}}, b \in P_{x'}^{\mathcal{H}}, \{a,b\} \in E(\mathcal{H})\big\}$
3. if $M(y) = \{x\}$, $M(y') = \{x'\}$, $x \neq x'$, and $\{x, x'\} \notin E(\mathcal{G}_X)$: $E_{\{y,y'\}} := \big\{\{v_a^y, v_b^{y'}\} \mid a \in P_x^{\mathcal{H}},\ b \in P_{x'}^{\mathcal{H}}\big\}$
4. if $M(y) = \{x\}$ and $M(y') = \{x, x'\}$: $E_{\{y,y'\}} := \big\{\{v_a^y, v_{\{a,b\}}^{y'}\} \mid a \in P_x^{\mathcal{H}},\ b \in P_{x'}^{\mathcal{H}}\big\}$

Finally, we set $E(\mathcal{H}^*) := \bigcup_{e \in E(\mathcal{G}_Y)} E_e$ and note that $\|\mathcal{H}^*\| = O(\|\mathcal{H}\|^2)$. For every homomorphism h from $\mathcal{G}_X^{\mathrm{id}}$ to $\mathcal{H}$ we define the mapping $h^* \colon Y \to V(\mathcal{H}^*)$ by

$$h^*(y) := \begin{cases} v_{h(x)}^y, & \text{if } M(y) = \{x\} \\ v_{\{h(x),h(x')\}}^y, & \text{if } M(y) = \{x, x'\} \end{cases}$$

The next claim provides the key property of our construction: h^* is a homomorphism from $\mathcal{G}_Y^{\mathrm{id}}$ to $\mathcal{H}^*$ and every homomorphism from $\mathcal{G}_Y^{\mathrm{id}}$ to $\mathcal{H}^*$ has this form, see the full version of this paper for a proof.

▷ Claim 23. $\mathrm{Hom}(\mathcal{G}_Y^{\mathrm{id}}, \mathcal{H}^*) = \{h^* : h \in \mathrm{Hom}(\mathcal{G}_X^{\mathrm{id}}, \mathcal{H})\}$

We finish the lemma by defining the mapping $\psi_{\mathcal{H}}$ that transforms any d-representation for $\mathrm{Hom}(\mathcal{G}_X^{\mathrm{id}}, \mathcal{H})$ into a d-representation for $\mathrm{Hom}(\mathcal{G}_Y^{\mathrm{id}}, \mathcal{H}^*)$. For each $x \in X$ we fix one $y_x \in Y$ such that $M(y_x) = \{x\}$ (those vertices exist by the definition of an almost minor map). Then we apply Lemma 5 and obtain a d-representation of $\pi_{\{y_x \colon x \in X\}} \mathrm{Hom}(\mathcal{G}_Y^{\mathrm{id}}, \mathcal{H}^*)$. After renaming every y_x to x and every v_a^y to a in the input labels of this circuit, we get a d-representation of $\mathrm{Hom}(\mathcal{G}_X^{\mathrm{id}}, \mathcal{H})$. ◀

With the following lemma we have everything in hand to proof our main theorem.

▶ **Lemma 24.** *For every k, $\mathcal{K}_k$ is an almost minor of $\mathcal{G}_{2k-2}$.*

[1] In particular the definition of a *depth-2 embedding* can be obtained from our definition of an almost minor by the following modifications. First remove clause (4). Second replace (2) with the following condition: for every $x \in X$ there is a $y \in Y$ s.t. $x \in M(y)$ and for every x, x' adjacent in $\mathcal{G}_X$ there exists *either* y, y' adjacent in $\mathcal{G}_Y$ such that $x \in M(y)$ and $x' \in M(y')$ *or* there exists y such that $\{x, x'\} \subseteq M(y)$. If we also remove clause (1) we get the general definition of an embedding.

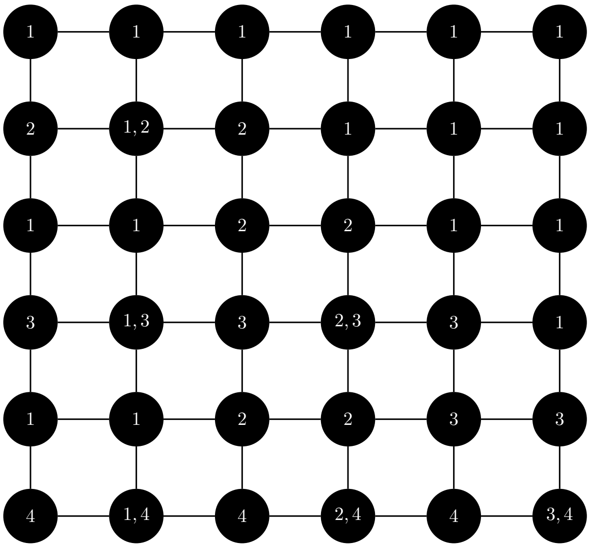

Figure 2 Construction from Lemma 24 for the case $k = 4$. The node in the ith row and jth column is labelled by the $\{a \mid u_a \in M(v_{i,j})\}$.

Proof of Lemma 24. Set

$$\begin{aligned} X :=& V(\mathcal{K}_k) = \{u_i \mid i \in [k]\},\\ Y :=& V(\mathcal{G}_{2k-2}) = \{v_{i,j} \mid i, j \in [2k-2]\}, \end{aligned}$$

where $v_{i,j}$ is the vertex in the ith row and jth column of the grid. Define $M \colon Y \to 2^X$ as follows:

1. if $j - 1 > i$, $M(v_{i,j}) = \{u_1\}$,
2. otherwise if $i \geq j - 1$ then:
   a. if i and j are both odd, $M(v_{i,j}) = \{u_{(j+1)/2}\}$,
   b. if i is odd and is j even, $M(v_{i,j}) = \{u_{j/2}\}$,
   c. if i is even and j is odd, $M(v_{i,j}) = \{u_{(i+2)/2}\}$,
   d. if i and j are both even, $M(v_{i,j}) = \{u_{(i+2)/2}, u_{j/2}\}$.

See Figure 2 for the case $k = 4$. It is easy to see that this map is almost minor, see the full version of this paper for a proof. ◀

Proof of Theorem 1. Let $\mathfrak{A}$ have unbounded treewidth. Then for every k there exists $\mathcal{B}_k \in \mathfrak{A}$ of treewidth at least $w(k)$. Then the Gaifman graph of $\mathcal{B}_k$, $\mathcal{G}_{\mathcal{B}_k}$ also has treewidth at least $w(k)$. By Theorem 20, $\mathcal{G}_{\mathcal{B}_k}$ has $\mathcal{G}_k$ as a minor. Since by Lemma 24, $\mathcal{K}_{(k+2/2)}$ is an almost minor of $\mathcal{G}_k$ we have:

$$\begin{aligned} \mathsf{d}_{\mathcal{B}_k}(m) &\underset{\text{(Lemma 14)}}{=} \Omega\left(\mathsf{d}_{\mathcal{B}_k^{\mathrm{id}},\mathfrak{C}}(m)\right)\\ &\underset{\text{(Lemma 15)}}{=} \Omega\left(\left(\mathsf{d}_{\mathcal{G}^{\mathrm{id}}_{\mathcal{B}_k},\mathfrak{H}}(m)\right)^{2/\,\mathsf{ar}(\mathcal{B}_k)}\right)\\ &\underset{\text{(Lemma 18)}}{=} \Omega\left(\left(\mathsf{d}_{\mathcal{G}^{\mathrm{id}}_k,\mathfrak{H}'}(m)\right)^{1/\,\mathsf{ar}(\mathcal{B}_k)}\right)\\ &\underset{\text{(Lemma 22)}}{=} \Omega\left(\left(\mathsf{d}_{\mathcal{K}^{\mathrm{id}}_{(k+2)/2},\mathfrak{D}}(m)\right)^{1/2\,\mathsf{ar}(\mathcal{B}_k)}\right) \end{aligned}$$

$$
\begin{aligned}
&\underset{\text{(Lemma 16)}}{=} \Omega\left(((\mathsf{d}_{\mathcal{K}_{(k+2)/2}}(m))^{1/2\,\mathsf{ar}(\mathcal{B}_k)}\right)\\
&\underset{\text{(Theorem 7)}}{=} \Omega\left(m^{(k+2)/4r} / \log^{(3k+2)/2r}(m)\right)
\end{aligned}
$$

Where $\mathfrak{C}$ is the class of $\sigma \cup \sigma_{B_k}$ structures $\mathcal{C}$ such that $\{P_a^{\mathcal{C}} \mid a \in B_k\}$ is a partition of the universe, $\mathfrak{H}$ the class of $V(\mathcal{G}_{\mathcal{B}_k}^{\mathrm{id}})$-partitioned graphs, $\mathfrak{H}'$ the class of $V(\mathcal{G}_k^{\mathrm{id}})$-partitioned graphs and $\mathfrak{D}$ the class of $V(\mathcal{K}_{(k+2)/2}^{\mathrm{id}})$-partitioned graphs. From the above we can conclude that (3) implies (1) in the statement of the theorem. Moreover, as discussed in Section 4, (1) implies (2) follows from [32] and (2) implies (3) trivially. ◀

7 Conclusion

Our main result characterises those bounded-arity classes of structures $\mathfrak{A}$ where the set of homomorphisms from $\mathcal{A} \in \mathfrak{A}$ to $\mathcal{B}$ can be succinctly represented. More precisely, the known upper bound of $O(\|\mathcal{A}\|^2\|\mathcal{B}\|^{\mathrm{tw}(\mathcal{A})+1})$ is matched by a corresponding lower bound of $\Omega(\|\mathcal{B}\|^{\mathrm{tw}(\mathcal{A})^{\varepsilon}})$, where $\mathrm{tw}(\mathcal{A})$ is the tree-width of $\mathcal{A}$ and $\varepsilon > 0$ is a constant depending on the excluded grid theorem and the arity of the signature. A future task would be to further close the gap between upper and lower bounds.

Another open question is to understand the representation complexity for all classes of structures $\mathfrak{A}$ (of unbounded arity). As mentioned in Section 4, a polynomial $O(\|\mathcal{A}\|^2 \cdot \|\mathcal{B}\|^{\mathrm{fhtw}(\mathcal{A})})$ upper bound was shown where $\mathrm{fhtw}(\mathcal{A})$ is the fractional hypertreewidth of $\mathcal{A}$ [32] and one might wonder whether this is tight. At least this is not the case in a parametrised setting, where a $f(\|\mathcal{A}\|)\|\mathcal{B}\|^{w}$ sized representation for some (not necessarily polynomial-time) computable f, is considered *tractable*. It is known that for structures $\mathcal{A}$ of *bounded submodular width* the homomorphism problem can be decomposed into a (not necessarily disjoint) union of $f(\|\mathcal{A}\|)$ instances of bounded fractional hypertreewidth [29, 6], leading to a d-representation of size $f(\|\mathcal{A}\|)\|\mathcal{B}\|^{\mathrm{subw}(\mathcal{A})}$ where $\mathrm{subw}(\mathcal{A})$ denotes the submodular width of $\mathcal{A}$, see Appendix A in the full version of this paper for details. Note that submodular width can be strictly smaller than fractional hypertreewidth [28]. For a more concrete example in this direction, the fractional hypertreewidth of $\mathcal{C}_4$ is 2, but one can show that $\mathrm{Hom}(\mathcal{C}_4, \mathcal{H})$ has deterministic d-representations of size $O(\|\mathcal{H}\|^{3/2})$ – almost matching the $O(\|\mathcal{H}\|^{3/2} / \log^7(\|\mathcal{H}\|))$ lower bound in Example 17. Note that while submodular width characterises the FPT-fragment of deciding the existence of homomorphisms on structures of unbounded arity [29], a tight characterisation for the parameterised counting problem is, despite some recent progress [23], still missing. In particular, it is not clear whether bounded submodular width implies tractable counting. We may face similar difficulties when studying the complexity of *deterministic* d-representations that allow efficient counting.

In the course of proving our main result we have developed tools and techniques for proving lower bounds on the size of d-representations, in particular using our k-clique lower bound as a starting point, defining an appropriate notion of reduction and showing that one can always get such a reduction if the "almost minor" relation holds. Whilst the proof of the clique lower bound in Section 5 exploits the specific nature of d-representations, we observe that much of the content of Section 6 can easily be used for other forms of representations. Since we now have understood the limitations of unrestricted d-representations, it would be good to know whether there are even more succinct representation formats that still allow efficient enumeration.

References

1 Noga Alon and Ravi B. Boppana. The monotone circuit complexity of boolean functions. *Comb.*, 7(1):1–22, 1987. doi:10.1007/BF02579196.

2 Antoine Amarilli, Pierre Bourhis, Louis Jachiet, and Stefan Mengel. A circuit-based approach to efficient enumeration. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 111:1–111:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.ICALP.2017.111.

3 Antoine Amarilli, Pierre Bourhis, Stefan Mengel, and Matthias Niewerth. Enumeration on trees with tractable combined complexity and efficient updates. In Dan Suciu, Sebastian Skritek, and Christoph Koch, editors, *Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2019, Amsterdam, The Netherlands, June 30 – July 5, 2019*, pages 89–103. ACM, 2019. doi:10.1145/3294052.3319702.

4 Jérôme Amilhastre, Hélène Fargier, Alexandre Niveau, and Cédric Pralet. Compiling CSPs: A Complexity Map of (Non-Deterministic) Multivalued Decision Diagrams. *International Journal on Artificial Intelligence Tools*, 23(4), 2014. doi:10.1142/S021821301460015X.

5 Albert Atserias, Martin Grohe, and Dániel Marx. Size bounds and query plans for relational joins. *SIAM Journal on Computing*, 42(4):1737–1767, 2013. doi:10.1137/110859440.

6 Christoph Berkholz and Nicole Schweikardt. Constant delay enumeration with FPT-preprocessing for conjunctive queries of bounded submodular width. *arXiv preprint*, 2020. arXiv:2003.01075.

7 Markus Bläser, Balagopal Komarath, and Karteek Sreenivasaiah. Graph pattern polynomials. In Sumit Ganguly and Paritosh K. Pandya, editors, *38th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2018, December 11-13, 2018, Ahmedabad, India*, volume 122 of *LIPIcs*, pages 18:1–18:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.FSTTCS.2018.18.

8 Andrei A. Bulatov. The complexity of the counting constraint satisfaction problem. *J. ACM*, 60(5):34:1–34:41, 2013. doi:10.1145/2528400.

9 Andrei A. Bulatov. A dichotomy theorem for nonuniform CSPs. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science (FOCS 2017)*, pages 319–330, 2017. doi:10.1109/FOCS.2017.37.

10 Andrei A. Bulatov, Victor Dalmau, Martin Grohe, and Daniel Marx. Enumerating Homomorphisms. In Susanne Albers and Jean-Yves Marion, editors, *26th International Symposium on Theoretical Aspects of Computer Science*, volume 3 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 231–242, Dagstuhl, Germany, 2009. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.STACS.2009.1838.

11 Chandra Chekuri and Julia Chuzhoy. Polynomial bounds for the grid-minor theorem. *J. ACM*, 63(5):40:1–40:65, 2016. doi:10.1145/2820609.

12 FRK Chung, P Erdős, and Joel Spencer. On the decomposition of graphs into complete bipartite subgraphs. In *Studies in pure mathematics*, pages 95–101. Springer, 1983.

13 Nadia Creignou and Jean-Jacques Hébrard. On generating all solutions of generalized satisfiability problems. *RAIRO Theor. Informatics Appl.*, 31(6):499–511, 1997. doi:10.1051/ita/1997310604991.

14 Víctor Dalmau and Peter Jonsson. The complexity of counting homomorphisms seen from the other side. *Theor. Comput. Sci.*, 329(1-3):315–323, 2004. doi:10.1016/j.tcs.2004.08.008.

15 Adnan Darwiche and Pierre Marquis. A knowledge compilation map. *J. Artif. Intell. Res.*, 17:229–264, 2002. doi:10.1613/jair.989.

16 Christian Engels. Dichotomy theorems for homomorphism polynomials of graph classes. *J. Graph Algorithms Appl.*, 20(1):3–22, 2016. doi:10.7155/jgaa.00382.

17 Tomás Feder and Moshe Y Vardi. Monotone monadic snp and constraint satisfaction. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, pages 612–622, 1993.

18 Herbert Fleischner, Egbert Mujuni, Daniël Paulusma, and Stefan Szeider. Covering graphs with few complete bipartite subgraphs. *Theoretical Computer Science*, 410(21-23):2045–2053, 2009.

19 Gianluigi Greco and Francesco Scarcello. Structural tractability of enumerating CSP solutions. *Constraints An Int. J.*, 18(1):38–74, 2013. doi:10.1007/s10601-012-9129-8.

20 Martin Grohe. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *J. ACM*, 54(1), March 2007. doi:10.1145/1206035.1206036.

21 Martin Grohe and Dániel Marx. Constraint solving via fractional edge covers. *ACM Transactions on Algorithms (TALG)*, 11(1):1–20, 2014.

22 Stasys Jukna and Alexander S Kulikov. On covering graphs by complete bipartite subgraphs. *Discrete Mathematics*, 309(10):3399–3403, 2009.

23 Mahmoud Abo Khamis, Ryan R. Curtin, Benjamin Moseley, Hung Q. Ngo, Xuanlong Nguyen, Dan Olteanu, and Maximilian Schleich. Functional aggregate queries with additive inequalities. *ACM Trans. Database Syst.*, 45(4), December 2020. doi:10.1145/3426865.

24 Balagopal Komarath, Anurag Pandey, and Chengot Sankaramenon Rahul. Monotone arithmetic complexity of graph homomorphism polynomials. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPIcs*, pages 83:1–83:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ICALP.2022.83.

25 Frédéric Koriche, Jean-Marie Lagniez, Pierre Marquis, and Samuel Thomas. Compiling Constraint Networks into Multivalued Decomposable Decision Graphs. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 332–338, Buenos Aires, Argentina, 2015. AAAI Press. URL: http://dl.acm.org/citation.cfm?id=2832249.2832295.

26 Yuan Li, Alexander A. Razborov, and Benjamin Rossman. On the ac^0 complexity of subgraph isomorphism. *SIAM J. Comput.*, 46(3):936–971, 2017. doi:10.1137/14099721X.

27 Dániel Marx. Can you beat treewidth? In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 169–179. IEEE, 2007.

28 Dániel Marx. Tractable structures for constraint satisfaction with truth tables. *Theory of Computing Systems*, 48(3):444–464, 2011.

29 Dániel Marx. Tractable hypergraph properties for constraint satisfaction and conjunctive queries. *Journal of the ACM (JACM)*, 60(6):1–51, 2013.

30 Robert Mateescu and Rina Dechter. Compiling Constraint Networks into AND/OR Multi-valued Decision Diagrams (AOMDDs). In *Principles and Practice of Constraint Programming – CP 2006*, Lecture Notes in Computer Science, pages 329–343. Springer, Berlin, Heidelberg, September 2006. doi:10.1007/11889205_25.

31 Dhruv Mubayi and György Turán. Finding bipartite subgraphs efficiently. *arXiv preprint*, 2009. arXiv:0905.2527.

32 Dan Olteanu and Jakub Závodný. Size bounds for factorised representations of query results. *ACM Transactions on Database Systems (TODS)*, 40(1):1–44, 2015.

33 Knot Pipatsrisawat and Adnan Darwiche. New compilation languages based on structured decomposability. In Dieter Fox and Carla P. Gomes, editors, *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI 2008)*, pages 517–522. AAAI Press, 2008. URL: http://www.aaai.org/Library/AAAI/2008/aaai08-082.php.

34 Neil Robertson and P.D Seymour. Graph minors. v. excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 41(1):92–114, 1986. doi:10.1016/0095-8956(86)90030-4.

35 Benjamin Rossman. On the constant-depth complexity of k-clique. In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 721–730. ACM, 2008. doi:10.1145/1374376.1374480.

36 Benjamin Rossman. The monotone complexity of k-clique on random graphs. *SIAM J. Comput.*, 43(1):256–279, 2014. doi:10.1137/110839059.

37 Henning Schnoor and Ilka Schnoor. Enumerating all solutions for constraint satisfaction problems. In Wolfgang Thomas and Pascal Weil, editors, *STACS 2007, 24th Annual Symposium on Theoretical Aspects of Computer Science, Aachen, Germany, February 22-24, 2007, Proceedings*, volume 4393 of *Lecture Notes in Computer Science*, pages 694–705. Springer, 2007. doi:10.1007/978-3-540-70918-3_59.

38 Dmitriy Zhuk. A proof of the CSP dichotomy conjecture. *Journal of the ACM (JACM)*, 67(5):1–78, 2020.