Definability of Cai-Fürer-Immerman Problems in Choiceless Polynomial Time

Wied Pakusa[1], Svenja Schalthöfer[2], and Erkal Selman[3]

1 **Mathematical Foundations of Computer Science, RWTH Aachen University, Germany**
pakusa@logic.rwth-aachen.de
2 **Mathematical Foundations of Computer Science, RWTH Aachen University, Germany**
schalthoefer@logic.rwth-aachen.de
3 **Logic and Theory of Discrete Systems, RWTH Aachen University, Germany**
selman@informatik.rwth-aachen.de

Abstract

Choiceless Polynomial Time (CPT) is one of the most promising candidates in the search for a logic capturing Ptime. The question whether there is a logic that expresses exactly the polynomial-time computable properties of finite structures, which has been open for more than 30 years, is one of the most important and challenging problems in finite model theory.

The strength of Choiceless Polynomial Time is its ability to perform isomorphism-invariant *computations* over structures, using hereditarily finite sets as data structures. But, as it preserves symmetries, it is choiceless in the sense that it cannot select an arbitrary element of a set—an operation which is crucial for many classical algorithms. CPT can define many interesting Ptime queries, including (the original version of) the Cai-Fürer-Immerman (CFI) query. The CFI query is particularly interesting because it separates fixed-point logic with counting from Ptime, and has since remained the main benchmark for the expressibility of logics within Ptime.

The CFI construction associates with each connected graph a set of CFI-graphs that can be partitioned into exactly two isomorphism classes called odd and even CFI-graphs. The problem is to decide, given a CFI-graph, whether it is odd or even. In the original version, the underlying graphs are linearly ordered, and for this case, Dawar, Richerby and Rossman proved that the CFI query is CPT-definable. However, the CFI query over general graphs remains one of the few known examples for which CPT-definability is open.

Our first contribution generalises the result by Dawar, Richerby and Rossman to the variant of the CFI query where the underlying graphs have colour classes of logarithmic size, instead of colour class size one. Secondly, we consider the CFI query over graph classes where the maximal degree is linear in the size of the graphs. For these classes, we establish CPT-definability using only sets of small, constant rank, which is known to be impossible for the general case.

In our CFI-recognising procedures we strongly make use of the ability of CPT to create sets, rather than tuples only, and we further prove that, if CPT worked over tuples instead, no such procedure would be definable. We introduce a notion of "sequence-like objects" based on the structure of the graphs' symmetry groups, and we show that no CPT-program which only uses sequence-like objects can decide the CFI query over complete graphs, which have linear maximal degree. From a broader perspective, this generalises a result by Blass, Gurevich, and van den Bussche about the power of isomorphism-invariant machine models (for polynomial time) to a setting with counting.

1998 ACM Subject Classification F.4.1 Mathematical Logic

Keywords and phrases finite model theory, descriptive complexity, logic for Ptime, Choiceless Polynomial Time, Cai-Fürer-Immerman

Digital Object Identifier 10.4230/LIPIcs.CSL.2016.19

25th EACSL Annual Conference on Computer Science Logic (CSL 2016).
Editors: Jean-Marc Talbot and Laurent Regnier; Article No. 19; pp. 19:1–19:17

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

One of the most important questions in descriptive complexity theory is whether there is a logic capturing Ptime [7, 14, 15].

Currently, there are two main branches of research approaching that problem. The first sets off from a seminal result of Immerman and Vardi which shows that (least) fixed-point logic captures Ptime on *ordered* structures [16, 19]. This precise logical characterisation of polynomial time heavily relies on the presence of a linear order on the input structure: Unfortunately, there are many queries, including trivial counting properties, which cannot be defined in fixed-point logic without a linear order. Still, the Immerman-Vardi Theorem indicates that fixed-point logic is a reasonable starting point to search for a logic capturing polynomial time. Indeed, much research has focused on ways to extend fixed-point logic by new operators in order to capture larger and larger fragments of Ptime "from below".

The best-studied of such formalisms is the extension of fixed-point logic by a counting mechanism, called fixed-point logic with counting or FP+C. Immerman proposed FP+C as a candidate to capture polynomial time on all finite structures. Indeed, it turned out that FP+C is very powerful and can express a robust and natural fragment of polynomial time. For a recent survey on FP+C, see [8].

However, Cai, Fürer and Immerman [6] found a very clever construction which demonstrates that FP+C fails to capture Ptime on all structures. This Cai-Fürer-Immerman query has since served as an extremely valuable tool for analysing the expressive power of logics over finite structures, as well as the complexity of the graph isomorphism problem.

Besides FP+C, further extensions of fixed-point logic by much stronger operators have been studied in recent years. The most important such extension is Rank Logic, proposed by Dawar, Grohe, Holm and Laubner in [9]. It is still open whether certain variants of Rank Logic [13] capture polynomial time.

Whereas studying extensions of fixed-point logic can be seen as searching for a logic for Ptime "from below", the second approach attacks the problem "from above". That branch of research aims for a formalism of Ptime *computation* that operates on structures and is *isomorphism-invariant*. Indeed, isomorphism-invariance is the main difference between logics and machine models, according to the standard definition, due to Gurevich [15], of what would constitute a logic for Ptime. Choiceless Polynomial Time, introduced by Blass, Gurevich and Shelah [3], approaches Ptime in that sense, with the intuition that it resembles classical Ptime computation "as closely as possible" without violating isomorphism-invariance.

Choiceless Polynomial Time

As it is isomorphism-invariant, Choiceless Polynomial Time, in contrast to actual computation models, lacks arbitrary choice: instructions like "choose an arbitrary vertex" are not possible. Instead, CPT performs parallel computations on possibly nested, logically definable sets over domain elements, using ordinals for counting operations. More precisely, CPT is able to construct hereditarily finite sets over the input structure, allowing for higher-order data structures. Because of its rather algorithmic nature, we often refer to CPT-programs or -algorithms instead of CPT-formulas. Most characteristics of a logic capturing Ptime are rather easy to verify for CPT. The challenge is to determine whether it can express *all* Ptime queries. To date, CPT has not been separated from Ptime, and many queries proposed for that purpose have been shown to be expressible, such as:

- Any query definable in FP+C is also definable in CPT.
- On structures with sufficiently large padding, CPT captures PTIME [4, 17]. This already implies that CPT is a strict extension of FP+C.
- The Cai-Fürer-Immerman query is definable in CPT, even without counting, if the underlying graphs are linearly ordered [10].
- CPT captures PTIME on structures with bounded colour class size if the colour classes have Abelian automorphism groups. A subprocedure shows that solvability of *cyclic linear equation systems* is CPT-definable [1].

For a more detailed survey of some recent insights, see also [11].

These results illustrate the surprising expressive power of Choiceless Polynomial Time. However, there are some properties for which we do not know whether they can be expressed in CPT, and which might witness a separation from polynomial time.

Most importantly, it is open whether the Cai-Fürer-Immerman problem over general graphs can be expressed in Choiceless Polynomial Time. Since the CFI query has proven to be an extremely valuable benchmark for the expressive power of logics over finite structures, solving this question would significantly increase our understanding of the expressive power of CPT. The CFI query additionally gives rise to particularly simple instances of various more general, and arguably more important, queries for which we do not know whether they can be expressed in Choiceless Polynomial Time. In particular, it is open whether CPT can define the isomorphism problem for graphs of bounded degree or solve linear equation systems over finite fields. For both of these problems certain variants of the CFI query constitute a rather simple class of instances, which means that it should be easier to find a CPT-procedure for the CFI query first before trying to solve these much more sophisticated problems in CPT in general.

The Cai-Fürer-Immerman Query

The CFI query is a subproblem of the graph isomorphism problem. With each *connected* graph $G = (V, E)$, the CFI construction associates a set of CFI-graphs $\mathfrak{G}^T$ over G, for every $T \subseteq V$, by replacing each vertex v of G by a certain graph gadget. More precisely, each vertex $v \in V$ can either be replaced by an *even* gadget ($v \notin T$) or by an *odd* gadget ($v \in T$). Hence, the CFI construction associates with every graph G an exponential-sized set of CFI-graphs $\{\mathfrak{G}^T : T \subseteq V\}$. However, it turns out that, up to isomorphism, there are in fact only two different CFI-graphs for fixed G. Indeed, a pair of CFI-graphs $\mathfrak{G}^T$ and $\mathfrak{G}^S$ over G is isomorphic if, and only if, the parity of the number of odd gadgets is the same, i.e. if $|T| \equiv |S| \pmod 2$. The CFI query is to determine, given a CFI-graph $\mathfrak{G}^T$, the parity of $|T|$. We give a precise definition of the CFI construction in Section 3.

It is known that the CFI query is decidable in PTIME, but not definable in FP+C [6]. Furthermore, it is definable in Rank Logic [9] and in Choiceless Polynomial Time without counting in case the underlying graph is *linearly ordered*, but not definable in CPT using only sets of bounded rank [10].

In the original work, Cai, Fürer and Immerman applied their construction to *ordered, three-regular* graphs. The effect is that the resulting CFI-graphs are also three-regular, and have colour class size four. Hence, Cai, Fürer and Immerman not only demonstrated that FP+C fails to capture polynomial time, but also that it cannot decide the isomorphism problem for graphs with bounded degree *and* bounded colour class size, though there are efficient isomorphism tests for both classes.

The immediate question is whether Choiceless Polynomial Time, as an extension of FP+C, can define these isomorphism problems and, in particular, whether it can distinguish

between the odd and even version of CFI-graphs. The first positive result in this context was achieved by Blass, Gurevich and Shelah [4], who proved that the Cai-Fürer-Immerman query can be expressed in CPT if the graphs come with a suitable padding, which makes it possible to define all linear orders on the input structure in spite of the space bound. This observation also lead to the separation of CPT and FP+C. However, it remained a challenging open question whether CPT can also define the CFI query without padding.

By using a very elegant construction, Dawar, Richerby and Rossman [10] were able to confirm this later for the special case where the underlying graphs are linearly ordered (as it is the case in the original construction of Cai, Fürer and Immerman). Essentially, the approach of Dawar, Richerby and Rossman is to succinctly represent the complete isomorphism class of a given CFI-graph—a class of exponential size—as a sufficiently small hereditarily finite set over the input structure. To this end they invent a data structure which uses highly nested sets. They further prove that this nesting cannot be avoided: with sets of constant rank it is not possible to define the CFI query over ordered graphs in CPT. Having a representation of the isomorphism class as a single object, they then show how to obtain, in CPT, a canonical numerical invariant for the given isomorphism class from which one can read off the parity of the CFI-graph. A more detailed description of this algorithm is given in Section 4.

In this paper, we set out to identify new classes of graphs over which the CFI query is CPT-definable. Moreover, we analyse the resources which are required by CPT-programs to decide the CFI query. Our first contribution is to generalise the result of Dawar, Richerby and Rossman from linearly ordered graphs to graphs with colour classes of *logarithmic* size, that is graphs with a built-in linear preorder on the universe which may contain classes of incomparable elements, but where the size of such classes is bounded logarithmically in the size of the input structure. This allows us to use all subsets of every colour class in a CPT-program. Note that the case of ordered graphs appears as the special case of colour classes of size one. Our procedure is based on the insight that it is possible to represent all linear orders which are consistent with the given preorder using polynomial resources if we iteratively join orderings defined over the same *set* of elements. The proof can be found in Section 5.

Secondly, we strengthen a claim from [10] for complete graphs and show that the CFI query over classes of *unordered* graphs where the maximal degree is linear is CPT-definable using only sets of constant rank. Recall that this is not possible in the general case.

If the underlying graph over n vertices has at least one vertex of degree $\frac{n}{k}$, the associated CFI-graph is of size $\geq 2^{\frac{n}{k}}$. Hence, a CPT-program can access *all subsets* of the vertex set of the underlying graph within the polynomial resource bounds. We use these subsets to inductively represent partial computations for the parity of the CFI-graph. Our construction is presented in Section 6.

Note that we *cannot* access all $n!$ different linear orderings of the underlying graphs with any resource bound polynomial in 2^n. Intuitively, this means that the power to use *sets* as abstractions of the linear orderings defined on their domain is vital for our CPT-procedure to respect polynomial bounds.

In Section 7 we use this observation to answer the following question: are isomorphism-invariant computation models with polynomial time bounds strictly more powerful if they use sets instead of only tuples? In the absence of counting, this question was answered by Blass, Gurevich and van den Bussche in [5]: set-like data structures are more powerful than sequence-like data-structures (see also [2].) We generalise this result to the case of counting and show that CPT-programs which decide the CFI query over complete graphs have to use set-like objects. In particular, this yields an interesting lower bound for a fragment of

Interpretation Logic, a characterisation of Choiceless Polynomial Time which was presented in [12]: Interpretation Logic without congruences cannot decide the Cai-Fürer-Immerman query over complete graphs, and is thus strictly less expressive than the fragment of CPT using only sets of bounded rank.

2 Choiceless Polynomial Time

Choiceless Polynomial Time is a polynomial time restriction of BGS logic (named after Blass, Gurevich and Shelah). The definition used here is based on the concise definition given by Rossman [18]. Let τ be a relational signature and let $\mathfrak{A} = (A, \tau)$ be a finite τ-structure. BGS logic is evaluated over the *hereditarily finite expansion* $\mathrm{HF}(\mathfrak{A})$ of $\mathfrak{A}$. The signature of $\mathrm{HF}(\mathfrak{A})$ is $\tau^{\mathrm{HF}} = \tau \uplus \{\emptyset, \mathrm{Atoms}, \in, \mathrm{Pair}, \mathrm{Union}, \mathrm{Unique}, \mathrm{Card}\}$, where $\emptyset$ and Atoms are constants, $\in$ is a binary relation symbol, Union, Unique and Card are unary function symbols and Pair is a binary function symbol. The domain of $\mathrm{HF}(\mathfrak{A})$ is the set of hereditarily finite sets over the *atoms* A. The *hereditarily finite sets* $\mathrm{HF}(A)$ are defined recursively as the finite sets of atoms and all finite sets of hereditarily finite objects, where an *object* is an atom or a set. In $\mathrm{HF}(\mathfrak{A})$, the relations and functions are interpreted as follows: $\emptyset^{\mathrm{HF}(\mathfrak{A})} = \emptyset$, $\mathrm{Atoms}^{\mathrm{HF}(\mathfrak{A})} = A$, $\in^{\mathrm{HF}(\mathfrak{A})} = \{(a, b) \mid a \in b\}$, $\mathrm{Pair}^{\mathrm{HF}(\mathfrak{A})}(a, b) = \{a, b\}$, $\mathrm{Union}^{\mathrm{HF}(\mathfrak{A})}(a) = \{b \in c : c \in a\}$, $\mathrm{Unique}^{\mathrm{HF}(\mathfrak{A})}(a) = b$ if $a = \{b\}$ for some b and $\emptyset$ otherwise, and $\mathrm{Card}^{\mathrm{HF}(\mathfrak{A})}(a) = |a|$ encoded as a von Neumann ordinal if a is a set and $\emptyset$ if $a \in A$.

BGS logic is defined by means of *terms*, *formulas* and *programs*. Terms are interpreted by values in $\mathrm{HF}(A)$. Syntactically, terms are variables, constants from τ^{HF}, objects $f\bar{t}$ for a function symbol $f \in \tau^{\mathrm{HF}}$ and terms $\bar{t}$, and *comprehension terms*. A comprehension term is of the form $t = \{s(\bar{x}, y) : y \in r(\bar{x}) : \varphi(\bar{x}, y)\}$, where s and r are terms and φ is a formula, with the semantics $\{s^{\mathfrak{A}}(\bar{a}, b) : y \in r^{\mathfrak{A}}(\bar{a}) : \mathfrak{A} \models \varphi(\bar{a}, b)\}$ for an assignment $\bar{a}$ of $\bar{x}$. A *formula* is either $t_1 = t_2$ or $R\bar{t}$ for terms $\bar{t} = t_1 \dots t_r$ and r-ary $R \in \tau$, or constructed from formulas with the connectives $\wedge, \vee, \neg$. A BGS *program* is a triple $\Pi = (\Pi_{\mathrm{step}}, \Pi_{\mathrm{halt}}, \Pi_{\mathrm{out}})$ of a term Π_{step} and formulas $\Pi_{\mathrm{halt}}, \Pi_{\mathrm{out}}$. The *run* of Π on $\mathfrak{A}$ is the sequence $(a_i)_{i \geq 0}$ such that $a_0 = \emptyset$ and $a_{i+1} = \Pi_{\mathrm{step}}(a_i)$ for $i > 0$. Choose $\kappa \leq \omega$ maximal such that $\mathfrak{A} \not\models \Pi_{\mathrm{halt}}(a_i)$ for $i < \kappa$. If κ is finite, then $\mathfrak{A} \models \Pi$ if and only if $a_\kappa \models \Pi_{\mathrm{out}}$.

To obtain Choiceless Polynomial Time, a fragment of BGS contained in PTIME, the complexity of the run and the occurring sets, measured in terms of their transitive closure, is bounded polynomially. A set y is *transitive* if $x \subseteq y$ for each $x \in y$. The *transitive closure* $\mathrm{tc}(x)$ of a set x is the least transitive set y with $x \subseteq y$. A CPT *program* is a pair $\bar{\Pi} = (\Pi, p)$, where Π is a BGS program and $p : \mathbb{N} \to \mathbb{N}$ a polynomial. The run of $\bar{\Pi}$ on $\mathfrak{A}$ is the maximal initial segment ρ of the run of Π on $\mathfrak{A}$ such that the length of ρ is at most $p(|A|)$ and, for each a_i in ρ, $\mathrm{tc}(a_i) \leq p(|A|)$. By CPT, we denote the variant with the function Card defined previously, unless stated otherwise.

We extend the syntax of CPT by assuming that there are distinct constants $0, 1$, and that there are syntactic means to define *tuples* explicitly. Note that tuples can be encoded as an extension of the Kuratowski encoding of pairs, so this is indeed only a change of syntax. We denote tuples as objects $\langle a_1, \dots, a_k \rangle$.

3 Graphs and the Cai-Fürer-Immerman Construction

Let $G = (V, E)$ be an undirected graph with vertex set V and edge set E. For a vertex $v \in V$, $E(v)$ denotes the set of edges incident to v. If $V', V'' \subseteq V$, $E[V']$ is the set of edges in the subgraph induced by V', and (V', V'') denotes the (V', V'')-cut, i.e. all edges $\{v, w\}$ with $v \in V'$ and $w \in V''$.

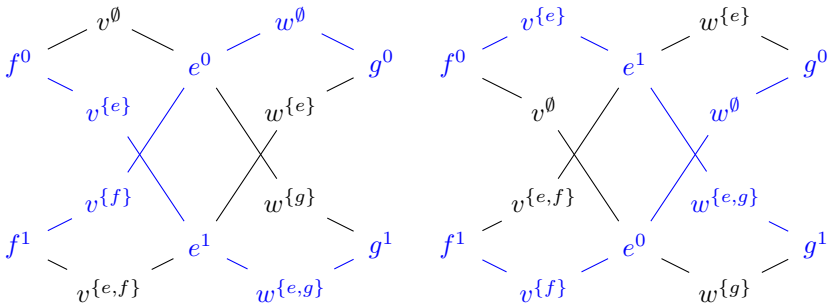

Figure 1 A subgraph of the graph $\mathfrak{G}$. The subgraph highlighted on the left corresponds to the CFI-graph $\mathfrak{G}^{\{v\}}$. The automorphism ρ_e maps $\mathfrak{G}^{\{v\}}$ to $\mathfrak{G}^{\{w\}}$, which is highlighted on the right.

A preordered graph $G^{\prec} = (V, E, \prec)$ is a graph $G = (V, E)$ equipped with a preorder $\prec$ on the vertices. For ease of notation, we identify the graph G with any given ordered version. A set of $\prec$-incomparable elements is called a *colour class*. We write C_i for the ith colour class in the linear order $\prec$ induces on the colour classes. A graph has colour classes of logarithmic size if $|C_i| \leq \log |V|$ for all colour classes C_i.

In the following, we present a definition and some properties of the Cai-Fürer-Immerman graphs. The CFI construction determines for each connected graph and each subset T of the vertex set a CFI-graph $\mathfrak{G}^T$. All $\mathfrak{G}^T$ fall into two isomorphism classes (*even* and *odd*). The CFI query then asks, given a CFI-graph, whether it is even or odd.

Given $G = (V, E)$, the CFI-graphs are certain subgraphs of the graph $\mathfrak{G}$ defined in the following. The graph $\mathfrak{G}$ contains, for each vertex $v \in V$, the *vertex gadget* $v^* = \{v^X : X \subseteq E(v)\}$ and, for each edge $e \in E$, the *edge gadget* $e^* = \{e^0, e^1\}$. The vertices v^X occurring in the vertex gadgets are called *inner vertices*. Let $\hat{V} = \cup_{v \in V} v^*$, $\hat{E} = \cup_{e \in E} e^*$, and for a subset $F \subseteq E$, $\hat{F} = \cup_{e \in F} e^*$. The edge set of $\mathfrak{G}$ is $\{\{v^X, e^1\} : v^X \in \hat{V}, e \in X\} \cup \{\{v^X, e^0\} : v^X \in \hat{V}, e \in E(v) \setminus X\}$.

Now let $T \subseteq V$. Then $v^*_T = \{v^X : |X| \text{ is odd}\}$ if $v \in T$ (then v^*_T is an odd gadget), $v^*_T = \{v^X : |X| \text{ is even}\}$ if $v \notin T$ (then v^*_T is even), and $\hat{V}_T = \cup_{v \in V} v^*_T$. The graph $\mathfrak{G}^T$ is the subgraph of $\mathfrak{G}$ induced by $\hat{E} \cup \bigcup_{v \in V} v^*_T$. We call $\mathfrak{G}^T$ even if $|T|$ is even, and odd otherwise. The construction is illustrated in Figure 1.

Consider an automorphism of $\mathfrak{G}$ that fixes v^* set-wise for each $v \in V$. Such an automorphism is always completely determined by a set $F \subseteq E$ such that for each $e \in F$, e^0 and e^1 are swapped. Formally, ρ_F is the mapping $e^i \mapsto e^{i-1}$ for $e \in F$ and $i \in \{0, 1\}$, $e^i \mapsto e^i$ for $e \notin F$, and $v^X \mapsto v^{X \triangle (F \cap E(v))}$. The group of automorphisms of $\mathfrak{G}$ that fix each v^* is generated by $\{\rho_e \coloneqq \rho_{\{e\}} : e \in E\}$.

As illustrated in Figure 1, each ρ_e maps $\mathfrak{G}^T$ to some $\mathfrak{G}^S$ such that T and S have the same parity. As shown by Cai, Fürer and Immerman [6], it follows that, for every connected graph G, $\mathfrak{G}^S \cong \mathfrak{G}^T$ if and only if $|S| \equiv |T| \pmod 2$. In other words, the even and odd CFI-graphs are uniquely determined up to isomorphism, and the automorphisms of $\mathfrak{G}$ stabilising all v^* are exactly the isomorphisms between graphs $\mathfrak{G}^T, \mathfrak{G}^S$ of the same parity.

The class of CFI-graphs over a graph class $\mathcal{C}$ is the class of all $\mathfrak{G}^T$ for $G \in \mathcal{C}$. If G is preordered by $\prec$, then $\prec$ induces an order on $\mathfrak{G}$ (and thus on all $\mathfrak{G}^T$) in the obvious way.

4 Computing the Parity of Cai-Fürer-Immerman Graphs over Ordered Graphs

A simple polynomial-time procedure for deciding the parity of a Cai-Fürer-Immerman graph just assigns to each vertex $e^i \in \hat{E}$ the label e^0 or e^1 and labels the vertices in $\hat{V}$ accordingly. By the nature of the automorphisms of $\mathfrak{G}$, this yields a graph $\mathfrak{G}^T$ that has the same parity as the input graph. Then T and thus the parity of $\mathfrak{G}^T$ can be determined easily.

A naive way to use that approach in CPT would require computing all the assignments of labels e^0 and e^1 in parallel and hence involve exponentially many sets, violating the polynomial resource bounds. However, the CPT procedure proposed by Dawar, Richerby and Rossman in [10] constructs an object which makes it possible to compute such an assignment in CPT using a linear order on the underlying graph.

Their construction represents, for each vertex v of the underlying graph, whether v is in T in the following way: τ_v contains the neighbourhoods of all inner vertices occurring in the given graph $\mathfrak{G}^T$, and $\tilde{\tau}_v$ contains the neighbourhoods of the remaining inner vertices of $\mathfrak{G}$. So each neighbourhood in τ_v contains an odd number of CFI vertices of the form e^1 if, and only if $v \in T$, if, and only if the number of e^1 in the neighbourhoods in $\tilde{\tau}_v$ is even.

Now consider an automorphism ρ of $\mathfrak{G}$ that maps $\mathfrak{G}^T$ to some $\mathfrak{G}^S$. Then, for each $v \in S \triangle T$, ρ swaps τ_v and $\tilde{\tau}_v$.

The aim is to compute the parity of $|T|$, which means to determine the number of τ_v whose elements contain an odd number of e^1 each, i.e. summing over all τ_v. Whenever an even number of τ_v is replaced by the respective $\tilde{\tau}_v$, the sum does not change, so, in particular, the sum is not affected by the automorphisms of $\mathfrak{G}$.

In order to compute the sum, the τ_v and $\tilde{\tau}_v$ are combined into sets $\mu_i, \tilde{\mu}_i$ for $i \leq |V|$ representing the parity of $|T|$ restricted to the first i vertices $v_1, \ldots, v_i$ (with respect to the linear order on V): $\mu_1 = \tau_{v_1}$, $\tilde{\mu}_1 = \tilde{\tau}_{v_1}$, $\mu_{i+1} = \{\langle \mu_i, \tau_{v_{i+1}} \rangle, \langle \tilde{\mu}_i, \tilde{\tau}_{v_{i+1}} \rangle\}$, and $\tilde{\mu}_{i+1} = \{\langle \mu_i, \tilde{\tau}_{v_{i+1}} \rangle, \langle \tilde{\mu}_i, \tau_{v_{i+1}} \rangle\}$. The algorithm computes the object $\mu_{|V|}$, which encodes the parity of $|T|$.

The $\mu_i, \tilde{\mu}_i$ can be viewed as representing sequences of $\tau_v, \tilde{\tau}_v$ such that the number of occurring $\tilde{\tau}_v$ is even in μ_i and odd in $\tilde{\mu}_i$. An automorphism of $\mathfrak{G}$ mapping $\mathfrak{G}^T$ to $\mathfrak{G}^S$ then changes an even number of τ_v to $\tilde{\tau}_v$ (since $S \triangle T$ is even) and thus can be shown to preserve $\mu_{|V|}$ and $\tilde{\mu}_{|V|}$. This property is called super-symmetry.

Super-symmetry means that whenever $e^0, e^1 \in e^*$ are swapped throughout the transitive closure of $\mu_{|V|}$, $\mu_{|V|}$ is fixed. Thus, assigning labels from $\{0, 1\}$ to a fixed edge gadget e^* in both possible ways in $\mu_{|V|}$ yields a unique result. These assignments can be computed sequentially using the linear order, resulting in a modified set where all neighbourhoods in τ_v contain an odd number of 1s if, and only if $v \in T$. From this object, the parity of $|T|$ can be computed.

5 Graphs with Colour Classes of Logarithmic Size

The algorithm from [10] processes the CFI-graph according to the linear order on the underlying graph. We generalise that construction by defining objects that correspond to multiple linear orders consistent with a given preorder, and obtain

▶ **Theorem 1.** *Let $\mathcal{K}$ be the class of connected, preordered graphs $G = (V, E, \prec)$ such that the size of each colour class is bounded by $\log |V|$. The CFI query over $\mathcal{K}$ is definable in Choiceless Polynomial Time.*

Without the linear order, there is no unique notion of the sets μ_i, $\tilde{\mu}_i$ described above that represent the parity of $|T \cap \{v_1, \dots, v_i\}|$. Instead, we define sets $\mu_M, \tilde{\mu}_M$ encoding the parity of $|T \cap M|$ for more general subsets $M \subseteq V$. The number of considered subsets M is kept small using the preorder. First, we construct objects for all subsets of the first colour class C_1, and obtain sets μ_{C_1} and $\tilde{\mu}_{C_1}$ for the full colour class. These are combined with all subsets of the second colour class, which yields $\mu_{C_1 \cup C_2}$, and so on for the remaining colour classes.

But the encoding of the μ_M, $\tilde{\mu}_M$ should again represent adding the value of a specific τ_v to a previous value. Therefore, for each subset $M \subseteq V$ for which μ_M and $\tilde{\mu}_M$ are constructed, all ways to decompose M as $M = N \uplus \{v\}$ for some v in the current colour class are considered.

To replace the e^i with their labels from $\{0, 1\}$, the algorithm in [10] also uses the linear order on the vertices. Because this is not possible with only a preorder, we already assign labels to the edge gadgets during the construction of the μ_M, $\tilde{\mu}_M$. More precisely, when viewing M as $N \uplus \{v\}$, all edges in the cut $(N, \{v\})$ are processed. So our CPT algorithm does not actually compute the μ_M and $\tilde{\mu}_M$, but modified objects ν_M, $\tilde{\nu}_M$ where all e^i for $e \in E[M]$ in the subgraph induced by M are already processed. Finally, the parity of T is extracted from the object ν_V using a parity function p.

Next, we describe these steps in detail. The sets τ_v, $\tilde{\tau}_v$ for $v \in V$ are defined as in [10], with one difference: To allow for assigning labels 0 or 1 to several vertices at once later without losing information, we equip each neighbourhood with a parity check bit, which will encode the parity of the number of edges that are labelled 1.

▶ **Definition 2.** Let $v \in V$. Then $\tau_v^T = \{\langle N(v^X), 0\rangle : v^X \in v_T^*\}$ and $\tilde{\tau}_v^T = \{\langle N(v^X), 0\rangle : v^X \in v^* \setminus v_T^*\}$, where $N(v^X)$ is the neighbourhood of v^X in the full graph $\mathfrak{G}$.

We omit the superscript T whenever it is clear from the context.

The τ_v, $\tilde{\tau}_v$ are combined to obtain objects μ_M, $\tilde{\mu}_M$ for certain subsets $M \subseteq V$ consistent with the preorder.

▶ **Definition 3.** Let $\mathcal{M}_i$ be the set of all $M \subseteq V$ such that $\bigcup_{j<i} C_j \subseteq M$ and $M \subseteq \bigcup_{j \leq i} C_j$. Then let $\mathcal{M}$ be the set of all $M \subseteq V$ that are in $\mathcal{M}_i$ for some i.

For subsets $M \in \mathcal{M}$, the objects $\mu_M, \tilde{\mu}_M$ are constructed inductively as follows.

▶ **Definition 4.** Let $T \subseteq V$, $v \in V$, $M \in \mathcal{M}$ and $v \notin M$. Then $\mu_{\{v\}}^T = \tau_v^T$, $\tilde{\mu}_{\{v\}}^T = \tilde{\tau}_v^T$, $\mu_{M,v}^T = \{\langle \mu_M^T, \tau_v^T\rangle, \langle \tilde{\mu}_M^T, \tilde{\tau}_v^T\rangle\}$, $\tilde{\mu}_{M,v}^T = \{\langle \mu_M^T, \tilde{\tau}_v^T\rangle, \langle \tilde{\mu}_M^T, \tau_v^T\rangle\}$, and, for $|M| > 1$, $\mu_M^T = \{\mu_{N,w}^T : N \in \mathcal{M} \text{ and } N \uplus \{w\} = M\}$ and $\tilde{\mu}_M^T = \{\tilde{\mu}_{N,w}^T : N \in \mathcal{M} \text{ and } N \uplus \{w\} = M\}$.

Like the μ_i^T, $\tilde{\mu}_i^T$ in the procedure by Dawar, Richerby and Rossman, the μ_M^T, $\tilde{\mu}_M^T$ characterise the parity of $|T \cap M|$, which can be shown by an easy induction on $|M|$.

▶ **Lemma 5.** $\mu_M^T = \mu_M^S \Leftrightarrow \tilde{\mu}_M^T = \tilde{\mu}_M^S \Leftrightarrow |S \cap M| \equiv |T \cap M| \pmod 2$ *and*
$\mu_M^T = \tilde{\mu}_M^S \Leftrightarrow \tilde{\mu}_M^T = \mu_M^S \Leftrightarrow |S \cap M| \not\equiv |T \cap M| \pmod 2$.

Next we define the objects ν_M, $\tilde{\nu}_M$ obtained by labelling the edge gadgets corresponding to the edges in the cut $(N, \{v\})$ when combining an object for a smaller set N with τ_v or $\tilde{\tau}_v$. For fixed v, a sufficiently small set of such mappings is determined by the vertices in v^*: Each v^X defines through its neighbourhood a unique vertex from each adjacent edge gadget. This allows to define, for each pair v^X, M, a mapping labelling edge vertices with binary values and aggregating these values in the parity check bit.

▶ **Definition 6.** Let $v^X \in \hat{V}$ and $M \subseteq V$. Then, for each set z and $i \in \{0, 1\}$, let $B_{v^X,M}(\langle z, i\rangle) = \langle z \setminus (\widehat{M, \{v\}}), j\rangle$, where $j = i + |(\widehat{M, \{v\}}) \cap N(v^X)| \mod 2$, and, for any other set y, $B_{v^X,M}(y) = \{B_{v^X,M}(z) : z \in y\}$.

Recall that $\widehat{(M, \{v\})}$ denotes the set of vertices in edge gadgets corresponding to edges in the cut $(M, \{v\})$.

Like the mappings from [10] that label the e^i by i or $1-i$ for a single edge $e \in E$, the $B_{v^X,M}$ are computed simultaneously for all $v^X \in v^*$. By similar symmetry arguments, this yields a unique object.

▶ **Lemma 7.** *If $z \in \mathrm{HF}(\hat{E})$ is fixed by the automorphisms ρ_e for all $e \in (M, \{v\})$, then $B_{v^X,M}(z) = B_{v^Y,M}(z)$ for any $X, Y \subseteq E(v)$ and $B_{v^X,M}(z)$ is fixed by the same automorphisms.*

Proof. Let $X' = X \cap (M, \{v\})$ and $Y' = Y \cap (M, \{v\})$. Since $X' \triangle Y' \subseteq (M, \{v\})$, $z = \rho_{X' \triangle Y'}(z)$. By definition of $B_{v^X,M}$ and $B_{v^Y,M}$, $B_{v^X,M}(\rho_{X' \triangle Y'}(z)) = B_{v^Y,M}(z)$. Thus $B_{v^X,M}(z) = B_{v^X,M}(\rho_{X' \triangle Y'}(z)) = B_{v^Y,M}(z)$. Obviously ρ_e fixes $B_{v^X,M}(z)$ for $e \in (M, \{v\})$, because the corresponding CFI vertices do not occur in $B_{v^X,M}(z)$. ◀

The previous lemma justifies writing $B_{v,M}(z)$ for the mapping removing edges along the cut (M, v) if z is fixed by suitable automorphisms. Using these mappings, the objects ν_M, $\tilde{\nu}_M$ can be defined. The sets ν, $\tilde{\nu}$ for sets $\{v\}$ or pairs M, v are defined like the corresponding sets $\mu, \tilde{\mu}$, where μ is now replaced by ν. The only change occurs when aggregating ν_M and $\tilde{\nu}_M$: Then the corresponding mapping is applied to the $\nu_{N,v}$ and $\tilde{\nu}_{N,v}$

▶ **Definition 8.** Let $M \in \mathcal{M}$ with $|M| > 1$. $\nu_M = \{B_{v,N}(\nu_{N,v}) : N \uplus \{v\} = M \text{ and } N \in \mathcal{M}\}$ and $\tilde{\nu}_M = \{B_{v,N}(\tilde{\nu}_{N,v}) : N \uplus \{v\} = M \text{ and } N \in \mathcal{M}\}$.

To show that the $\nu_M, \tilde{\nu}_M$ are well-defined, we have to show that the construction only uses the mapping $B_{v,M}$ for sets that are fixed by all necessary automorphisms.

The proof uses that, like $\mu^T_M, \tilde{\mu}^T_M$, the sets $\nu^T_M, \tilde{\nu}^T_M$ characterise the parity of $|T \cap M|$. To show this, and make the $\nu^T_M, \tilde{\nu}^T_M$ more accessible to further analysis, we show that ν_M (resp. $\tilde{\nu}_M$) arises from μ_M (resp. $\tilde{\mu}_M$) by labelling and removing all edges in $E[M]$. That notion is formalised via mappings labelling arbitrary (sets of) edge gadgets in a way that is not CPT-definable in general, but makes it possible to reason about a specific replacement. We then show that, on the objects $\nu_M, \tilde{\nu}_M$, both kinds of mappings have the same effect.

▶ **Definition 9.** Let $e \in E$. Then $B_e(\langle z, i\rangle) = \langle z \setminus \{e\}, i + j\rangle$, where $j = 1$ if and only if $e^1 \in z$ and $+$ is addition modulo 2. For any other set y, $B_e(y) = \{B_e(z) : z \in y\}$. Let $F \subseteq E$ and let $e_1, \ldots, e_k$ be an enumeration of F. Then $B_F = B_{e_1} \circ \cdots \circ B_{e_k}$. Note that, for any $e \neq e'$, $B_e \circ B_{e'} = B_{e'} \circ B_e$, so B_F is well-defined.

Now the sets $\nu_M, \tilde{\nu}_M$ can be characterised as follows.

▶ **Lemma 10.** *Let $S, T \subseteq V$, $v \in V$ and $M, N \in \mathcal{M}$ such that $N \uplus \{v\} = M$.*

1. **(a)** $\nu^T_{N,v} = \nu^S_{N,v} \Leftrightarrow \tilde{\nu}^T_{N,v} = \tilde{\nu}^S_{N,v} \Leftrightarrow |T \cap M| \equiv |S \cap M| \pmod 2$,
   (b) $\tilde{\nu}^T_{N,v} = \nu^S_{N,v} \Leftrightarrow \nu^T_{N,v} = \tilde{\nu}^S_{N,v} \Leftrightarrow |T \cap M| \not\equiv |S \cap M| \pmod 2$.
2. *$\nu_{N,v}$, $\tilde{\nu}_{N,v}$ are fixed by the automorphism ρ_e for every $e \in (N, \{v\})$.*
3. *$\nu_M = B_{E[M]}(\tilde{\mu}_M)$ and $\tilde{\nu}_M = B_{E[M]}(\mu_M)$.*
4. **(a)** $\nu^T_M = \nu^S_M \Leftrightarrow \tilde{\nu}^T_M = \tilde{\nu}^S_M \Leftrightarrow |T \cap M| \equiv |S \cap M| \pmod 2$,
   (b) $\nu^T_M = \tilde{\nu}^S_M \Leftrightarrow \tilde{\nu}^T_M = \nu^S_M \Leftrightarrow |T \cap M| \not\equiv |S \cap M| \pmod 2$.

Proof. Simultaneously by induction on $|M|$. ◀

▶ **Corollary 11.** *ν_M is well-defined for all $M \in \mathcal{M}$.*

So the final set ν_V is a hereditarily finite set over $\{0, 1\}$ representing the parity of $|T|$. To extract that parity, we use the following aggregation function on $\mathrm{tc}(\nu_V)$:

▶ **Definition 12.**

$$p(x) = \begin{cases} i, & x = \langle N(v^X), i\rangle, \\ p(x_1) + p(x_2) \mod 2, & x = \langle x_1, x_2\rangle \text{ and } x_2 \notin \{0,1\}, \\ \prod_{y\in x} p(y), & x \text{ is a set.} \end{cases}$$

▶ **Lemma 13.** *$p(\nu_V^T) = 0$ if and only if $|T|$ is even.*

Proof. As, by Lemma 10, $\nu_V^T = B_E(\mu_V^T)$ and B_E labels the edges throughout the transitive closure, we can reason inductively about objects for smaller sets where *all* edges have been removed. Note that it suffices to consider the cases $T = \emptyset$ and $T = \{x\}$ for some $x \in V$. Thus the lemma is shown if the following statements are verified for all $M \subseteq V$ and $v \in V \setminus M$, where $P(x)$ denotes $p(B_E(x))$:

- $P(\mu_M^\emptyset) = P(\mu_{M,v}^\emptyset) = 0$, $P(\tilde{\mu}_M^\emptyset) = P(\tilde{\mu}_{M,v}^\emptyset) = 1$,
- $P(\mu_{M,v}^{\{x\}}) = 1 \Leftrightarrow P(\tilde{\mu}_{M,v}^{\{x\}}) = 0 \Leftrightarrow x \in M \cup \{v\}$,
- $P(\mu_M^{\{x\}}) = 1 \Leftrightarrow P(\tilde{\mu}_M^{\{x\}}) = 0 \Leftrightarrow x \in M$.

The statements can be shown by induction on $|M|$, and follow from the definition of the objects $\tau_v^T, \tilde{\tau}_v^T, \mu_M^T, \tilde{\mu}_M^T$ and the mappings B_E and p. ◀

It remains to show that the construction is CPT-definable. All sets used in the computation can be defined by the set-theoretic operations available in CPT. Thus we only need to prove that the construction does not use too many or too complex objects.

▶ **Lemma 14.** *$|\operatorname{tc}(\mu_M^T)|$ is polynomial in $|\mathfrak{G}^T|$ for $M, T \subseteq V$.*

Proof. First note that the sets $\mu_M, \tilde{\mu}_M$ for $|M| = k+1$ are constructed from objects created for sets of size k, so it suffices to count the newly created sets for $|M| = k+1$. The sets $\mu_{M,v}$, $\tilde{\mu}_{M,v}$ are constructed for at most 2^{C_i} many sets and $|C_i|$ many vertices for some colour class C_i of logarithmic size. Furthermore, there are also at most $2^{|C_i|}$ many new sets μ_M and $\tilde{\mu}_M$, which are built from $\mu_{M,v}$ and $\tilde{\mu}_{M,v}$. ◀

By Lemma 10, $|\operatorname{tc}(\nu_M)| \leq |\operatorname{tc}(\mu_M)|$ for all $M \subseteq V$. So our construction is CPT-definable, which completes the proof of Theorem 1.

6 Classes of Unordered Graphs with Linear Maximal Degree

The techniques used for graphs with colour classes of logarithmic size can be further refined for classes of graphs that do not possess any order, but where the maximal degree is linear in the size of the graph. Note that this implies that the CFI-query over, for example, complete graphs is CPT-definable, verifying a claim from [10].

▶ **Theorem 15.** *For every $k \in \mathbb{N}$, the CFI query over the graphs $G = (V, E)$ with maximal degree $\geq \frac{|V|}{k}$ is definable in* CPT *using only sets of constant rank not depending on k.*

We now explain the modifications made to adapt the procedure presented in the previous section to these graph classes.

As previously, we start with an object τ_v for each $v \in V$, that encodes whether $v \in T$. In the case of logarithmic colour classes, the number of subsets for which the "sum" of the τ_v is computed is kept small by using the preorder. The set of these subsets can be defined in CPT. For unordered graphs, CPT cannot distinguish between any two equally sized subsets of the vertex set. However, the CFI-graphs graphs are large enough to permit considering *all*

subsets of V, because for the vertex v with maximal degree, each v^X corresponds to a subset of v's $\geq \frac{|V|}{k}$ many neighbours.

Recall that, in the algorithm in [10], sequences of τ_v and $\tilde{\tau}_v$ with an even number of $\tilde{\tau}_v$ are encoded. With access to all subsets of V, these sequences can be replaced by all sets of τ_v and $\tilde{\tau}_v$ containing an even number of $\tilde{\tau}_v$. The sets μ_M, $\tilde{\mu}_M$ for $M \subseteq V$ defined in that way again characterise the parity of $|T \cap M|$.

We again use the mappings $B_{v,M}$ arising from Lemma 7 to remove edges along cuts (M, v). However, these mappings require a vertex v to be fixed during the construction. So we formalise the construction of the μ_M, $\tilde{\mu}_M$ as successively enlarging the sets M by a single vertex. For instance, whenever $M = N \uplus \{v\}$, each set in μ_M containing an even number of $\tilde{\tau}_w$ can be constructed by adding τ_v to a set in μ_N, or $\tilde{\tau}_v$ to a set in $\tilde{\mu}_M$. Since the choice of N and v does not affect the resulting sets, μ_M can be computed by simultaneously using all these decompositions of M.

So the sets ν_M with labelled edges are constructed by splitting M into all possible decompositions $N \uplus \{v\}$, adding τ_v (resp. $\tilde{\tau}_V$) to the sets in μ_N ($\tilde{\mu}_N$) and processing the edges along the cut $(N, \{v\})$.

Analogously to Lemma 10, we can show that the sets $\nu_M, \tilde{\nu}_M$ are well-defined, again using the mappings B_F from Definition 9, that remove edges in a specific set $F \subseteq E$. It also follows that ν_M is well-defined for all $M \subseteq V$.

The parity of $|T|$ is computed with a new aggregation function. It also first extracts and multiplies the parity bits from the neighbourhoods in each τ_v. Since the parity of the neighbourhoods in τ_v, $\tilde{\tau}_v$ determines whether v^* is an odd gadget, the product is 1 for τ_v if and only if v^* is odd. Next, the sum of the parities (modulo 2) of each set in ν_V consisting of an even number of $\tilde{\tau}_v$ is computed. By definition, that sum is even for every such set if and only if the number of odd vertex gadgets is even. So it is possible to extract the parity of $|T|$ from ν_V.

It remains to show that the ν_M can be constructed in Choiceless Polynomial Time. One of the main obstacles is to compute sums modulo 2 when applying the mappings $B_{v,M}$ and p, because counting would in general require ordinals, which do not have constant rank. Furthermore, we need to show that the size of the transitive closure of each ν_M is polynomial in $|\mathfrak{G}|$.

▶ **Lemma 16.** *For any definable set $x \in \mathrm{HF}(\mathfrak{A})$ of rank k with $|x|$ logarithmic in $|\mathfrak{A}|$, the parity of $|x|$ can be computed in* CPT *over $\mathfrak{A}$ with sets of rank k.*

First note that the size of each neighbourhood occurring in the transitive closure of each ν_M is bounded by $|V|$ and thus logarithmic in the size of the CFI-graph, so the lemma can be applied to the computation of the mappings $B_{v,M}$.

Proof. For all $n \leq |x|$, compute the set of all subsets of x of size n and store their parity. Start with the set of all singletons and parity 1. Given sets of size n, construct the sets of size $n+1$ analogously to the construction of the μ_M and flip the parity. As soon as the set $\{x\}$ itself has been constructed, the parity of $|x|$ can be extracted. ◀

▶ **Lemma 17.** *If G has a vertex of degree $\geq \frac{|V|}{k}$, ν_V^T can be constructed in Choiceless Polynomial Time from the input $\mathfrak{G}^T$.*

Proof. Each set ν_M or $\tilde{\nu}_M$ constructed by the algorithm contains $\leq 2^{|M|}$ many sets consisting of (polynomially sized) sets τ_v and $\tilde{\tau}_v$, and there are $2^{|V|}$ subsets M. The number and size of the ν_M and $\tilde{\nu}_M$ is polynomial in $|\mathfrak{G}^T|$ because some $v \in V$ has $\geq \frac{|V|}{k}$ neighbours, so the

vertex gadget v_T^* is of size $2^{\frac{|V|}{k}-1}$. So the transitive closure of all constructed objects is of polynomial size. ◀

7 Computations Are More Powerful Over Sets Than Over Tuples

In this section we want to show that every CPT-program which decides the Cai-Fürer-Immerman query over complete graphs (where the maximal degree is obviously linear) has to use set-like objects.

Of course, the immediate question is: what are set-like objects?

In order to motivate our following definition, let us consider a simple example. Let $G = (V, E)$ be a complete graph over a set V with $|V| = n$ vertices. The automorphism group of G is the full symmetric group $\Gamma = \mathrm{Sym}(V)$. Moreover, let $x = \{a_0, \dots, a_{k-1}\} \subseteq V$ be a *set* consisting of k distinct vertices, and let $y = (a_0, \dots, a_{k-1}) \in V^k$ be some arrangement of these k vertices as a *k-tuple* over V. Since every CPT-computation on G is invariant under the action of Γ, every CPT-program which constructs x also has to completely construct $\mathrm{Orbit}(x) = \{\pi(x) : \pi \in \Gamma\}$ of x (the same holds for every program which constructs y).

The question is, what do we know about the sizes of the orbits of x and y? It is easy to see that $|\mathrm{Orbit}(x)| = \binom{n}{k}$ while $|\mathrm{Orbit}(y)| = \binom{n}{k} \cdot k!$. Hence, for large k the orbit of the *set* x is much smaller than the size of the orbit of the *tuple* y (although they are defined over the same set of atoms). To put it differently, for large k the size of the stabiliser $\mathrm{Stab}(x)$ of x is much larger than the size of the stabiliser $\mathrm{Stab}(y)$ of y. Indeed, $|\mathrm{Stab}(x)| = k! \cdot (n-k)!$ and $|\mathrm{Stab}(y)| = (n-k)!$. Obviously, the reason is that in order to fix the tuple y we have to fix *every* entry a_i *point-wise* whereas we can permute the elements $\{a_0, \dots, a_{k-1}\}$ while keeping x fixed. In algebraic terms, the point-wise stabiliser $\mathrm{Stab}^\bullet\{a_1, \dots, a_k\} = \{\pi : \pi(a_i) = a_i \text{ for all } i < k\}$ of $a_0, \dots, a_{k-1}$ coincides with $\mathrm{Stab}(y)$ while it is a subgroup of $\mathrm{Stab}(x)$ of index $k!$.

▶ **Definition 18.** Let $\mathfrak{A}$ be a structure with automorphism group $\Gamma = \mathrm{Aut}(\mathfrak{A})$. Let $x \in \mathrm{HF}(A)$. A set $\sigma \subseteq A$ of atoms is a *support* for x if $\mathrm{Stab}^\bullet(\sigma) \leq \mathrm{Stab}(x)$, and σ is called a *strong support* of x if $\mathrm{Stab}^\bullet(\sigma) = \mathrm{Stab}(x)$.

Accordingly, we say that an element $x \in \mathrm{HF}(A)$ is *strongly supported* if it has a strong support. For example, every atom $a \in A$ is strongly supported and every $x \in \mathrm{HF}(A)$ with $\mathrm{Stab}(x) = \Gamma$ has the strong support $\emptyset$. Note that for the above example, the set $\{a_0, \dots, a_{k-1}\}$ is a support for the *set* $x = \{a_0, \dots, a_{k-1}\}$ and it is a *strong support* for the *tuple* $y = (a_0, \dots, a_{k-1})$. The notion of strongly supported sets is taken as a working definition for objects which are "not set-like", or more intuitively, which are "sequence-like". Those objects enforce an inherent order on the supporting atoms. We aim to prove Theorem 23: no CPT-program which can only access strongly supported objects can express the CFI query over complete graphs.

As a starting point for the analysis of strong supports, we examine the structure of the automorphism group of $\mathfrak{G}^T$.

▶ **Lemma 19** (Action of $\mathrm{Sym}(V)$)**.** *Let $G = (V, E)$ be a complete graph and let $S_V = \mathrm{Aut}(G) = \mathrm{Sym}(V)$. Then every $\pi \in S_V$ can be lifted to an automorphism $\rho \in \Gamma = \mathrm{Aut}(\mathfrak{G}^T)$ such that $\rho(v^*) = \rho(w^*)$ if and only if $\pi(v) = \pi(w)$ for all $v, w \in V$.*

Proof. Constructing ρ for a transposition $(v, w) \in S_V$, one has to make sure that all edge gadgets $\{u, v\}$ and $\{u, w\}$, and their neighbours in the vertex gadgets, are exchanged, and the parity of v^* and w^* is exchanged if necessary. ◀

The automorphism group $\Gamma = \mathrm{Aut}(\mathfrak{G}^T)$ of a CFI-graph over a complete graph $G = (V, E)$ can be decomposed as follows. Let us denote by Δ the subgroup of Γ which stabilises all sets of inner vertices v^* for $v \in V$, i.e. $\Delta = \bigcap_{v \in V} \mathrm{Stab}(v^*)$. Then Δ is a normal subgroup of Γ which can be identified with a subgroup of $\mathbb{Z}_2^{|E|}$ and which coincides with the CFI-automorphism group that one obtains if the underlying complete graph is ordered. It then follows, by the preceding lemma, that $\Gamma/\Delta \cong S_V$. We will often identify S_V with this group and also the corresponding actions of Γ/Δ on $\{v^* : v \in V\}$ and of S_V on V.

Next, we explain the idea of our proof, which is based on the techniques developed by Dawar, Richerby and Rossman in [10]. It is well-known that every CPT-program Π can be translated into a formula φ_Π of infinitary logic with counting and k variables, for a certain k which depends on Π (we denote this logic by $C^k_{\infty\omega}$), with the following property: for every input structure $\mathfrak{A}$, the formula φ_Π is equivalent to Π in the sense that

$$\mathfrak{A} \models \Pi \Leftrightarrow \mathrm{Active}(\Pi, \mathfrak{A}) \models \varphi_\Pi,$$

where $\mathrm{Active}(\Pi, \mathfrak{A})$ is the extension of $\mathfrak{A}$ by all hereditarily finite sets which are *activated* (intuitively: have to be constructed) during the run of Π on $\mathfrak{A}$. For details see [10, 3, 4]. Hence, if we can show for two structures $\mathfrak{A}$ and $\mathfrak{B}$ that $\mathrm{Active}(\Pi, \mathfrak{A})$ and $\mathrm{Active}(\Pi, \mathfrak{B})$ cannot be distinguished by any $C^k_{\infty\omega}$-formula ($\mathrm{Active}(\Pi, \mathfrak{A}) \equiv^C_k \mathrm{Active}(\Pi, \mathfrak{B})$), then we can conclude that Π cannot distinguish between $\mathfrak{A}$ and $\mathfrak{B}$ as well. However, there are two serious difficulties in this approach:

- Showing that $\mathrm{Active}(\Pi, \mathfrak{B}) \equiv^C_k \mathrm{Active}(\Pi, \mathfrak{B})$ is combinatorially extremely challenging, because we have to relate highly-nested sets over the two input structures.
- The structures $\mathrm{Active}(\Pi, \mathfrak{A})$ and $\mathrm{Active}(\Pi, \mathfrak{B})$ depend on the program Π. This is problematic as it would require to show $C^k_{\infty\omega}$-equivalence of structures which are defined rather indirectly by the run of the CPT-program Π.

There is a very nice approach, which was first used by Blass, Gurevich and Shelah in [3] and later refined and generalised by Dawar, Richerby and Rossman in [10, 18] which solves both problems at once. Very roughly, the idea is that for certain structures $\mathfrak{A}$ we can over-approximate $\mathrm{Active}(\Pi, \mathfrak{A})$ by the structure $\mathrm{HF}(\mathfrak{A})_\ell$ consisting of all hereditarily finite sets which are *transitively ℓ-supported*, that is those sets which have a support of size at most ℓ and whose transitive closure only consists of sets which again have a support of size at most ℓ. Specifically, if we can prove that *all* sets which can be activated in a run of Π on $\mathfrak{A}$ are ℓ-supported, then we know that $\mathrm{Active}(\Pi, \mathfrak{A}) \subseteq \mathrm{HF}(\mathfrak{A})_\ell$. More importantly, if the orbits of tuples (of sufficient length) in the structure $\mathfrak{A}$ are definable in $C^m_{\infty\omega}$ for a certain m ($\mathfrak{A}$ is then called C^m-homogeneous), then we can obtain a description of the possibly highly-nested transitively ℓ-supported sets in terms of their supports (which are lists of atoms) and syntactic objects which do not depend on $\mathfrak{A}$. This reduces the complexity to deal with highly nested sets in $\mathrm{HF}(\mathfrak{A})$ to flat objects over $\mathfrak{A}$. Our aim is to adapt the approach of Dawar, Richerby and Rossman (Theorem 22) for our application. To this end CFI-graphs over complete graphs should satisfy the following requirements:

- they are C^m-homogeneous for a certain m (and large enough tuples), this is Lemma 20, and
- all objects which can be activated by a CPT-program that only uses strongly supported sets have small supports (Lemma 21).

▶ **Lemma 20** (C^m-homogeneity). *Let $n, m \in \mathbb{N}$ such that $3 \leq m \leq n - 3$. Let $\mathfrak{G}^T$ be a CFI-graph over a complete graph $G = (V, E)$ with $|V| = n$. Let $\vec{a}$, $\vec{b}$ be tuples of vertices of $\mathfrak{G}^T$ both of length i for some $i \leq m - 3$. Then, $(\mathfrak{G}^T, \vec{a}) \equiv^C_m (\mathfrak{G}^T, \vec{b})$ implies $\mathit{Orbit}(\vec{a}) = \mathit{Orbit}(\vec{b})$.*

*In other words, CFI-graphs over complete graphs with n vertices are C^m-*homogeneous *(with respect to tuples of length $i \leq m-3$).*

▶ **Lemma 21** (Sizes of supports). *For all $q \geq 1$, $0 < \epsilon < 1$, we can find $m \geq 1$ such that for all strongly supported sets $x \in \mathrm{HF}(\hat{V}_T \cup \hat{E})$ over CFI-graphs $\mathfrak{G}^T$ of complete graphs $G = (V, E)$ with $|V| = n \geq m$ vertices we have*

- *$|Orbit(x)| > 2^{n \cdot q}$ or*
- *$|\sigma| \leq \epsilon \cdot n$ for some (not necessarily strong) support σ for x.*

In other words, all strongly supported sets $x \in \mathrm{HF}(\hat{V}_T \cup \hat{E})$ with small orbit $|\mathrm{Orbit}(x)| \leq 2^{n \cdot q}$ are $(\epsilon \cdot n)$-supported.

Proof. We assume that $|\mathrm{Orbit}(x)| \leq 2^{n \cdot q}$ and choose a strong support $\sigma \subseteq \hat{V}_T \cup \hat{E}$ for x which is minimal with respect to $\subseteq$. From the orbit stabiliser theorem, the fact that $S_V \leq \Gamma$ (Lemma 19), and since σ is a strong support for x it follows that

$$\frac{|S_V|}{|\mathrm{Stab}^{\bullet}_{S_V}(\sigma)|} \leq \frac{|\Gamma|}{|\mathrm{Stab}^{\bullet}_{\Gamma}(\sigma)|} = \frac{|\Gamma|}{|\mathrm{Stab}_{\Gamma}(x)|} = |\mathrm{Orbit}(x)| \leq 2^{n \cdot q}. \tag{1}$$

Our next aim is to show that the support σ induces a partition of V as $V = V_0 \uplus V_1 \uplus \cdots \uplus V_\ell$, such that every automorphism $\pi \in \mathrm{Stab}^{\bullet}_{S_V}(\sigma)$ respects this partition. More precisely, the following holds:

- $\mathrm{Stab}^{\bullet}_{S_V}(\sigma) \leq \mathrm{Stab}_{S_V}(\bigcup_{v \in V_0} v^*_T) \cap \cdots \cap \mathrm{Stab}_{S_V}(\bigcup_{v \in V_\ell} v^*_T)$, and
- $\ell = |\sigma \cap \hat{V}_T|$, and
- using the elements in σ as parameters one can define each of the sets V_i (inside the structure $\mathfrak{G}^T$) using a $C^{\ell+2}_{\infty\omega}$-formula.

To simplify our notation, we set $V^*_i = \bigcup_{v \in V_i} v^*_T$. To start, we first define $W \subseteq V$ as the set of vertices $v \in V$ such that σ contains an inner node from the CFI-gadget associated with the vertex v, that is $\sigma \cap v^*_T \neq \emptyset$. Then, clearly, $\mathrm{Stab}^{\bullet}_{S_V}(\sigma) \leq \bigcap_{w \in W} \mathrm{Stab}_{S_V}(\{w^*_T\}) \cap \mathrm{Stab}_{S_V}((V \setminus W)^*)$ and we obtain a first partition of V into $|W|+1$ many blocks as $V = \biguplus_{w \in W}\{w\} \uplus V \setminus W$.

For all $w \in W$, we choose a witnessing element $a_w \in \sigma \cap w^*_T$. We next want to show that whenever we fix an additional element $b \in \sigma \cap w^*_T$ for some $w \in W$ and $b \neq a_w$, then we obtain a refined partition of V which still satisfies the above properties. We show this by induction on the number $i = 0, \ldots, |\sigma \cap \hat{V}^T| - |W|$ of processed inner nodes which are different from the a_w. Thus, let us assume that we have already considered the elements in $\sigma_i = \{a_w, b_1, \ldots, b_i : w \in W\} \subseteq \sigma \cap \hat{V}^T$ and obtained a refined partition of V as $V = \biguplus_{w \in W}\{w\} \uplus V_1 \uplus \cdots \uplus V_i \uplus V \setminus (\bigcup_{j=1}^{i} V_j \cup W)$. Choose $b \in (\sigma \cap w^*_T) \setminus \sigma_i$ for some $w \in W$. Then $a_w = w^X$ and $b = w^Y$ for some sets $X, Y \subseteq E(w)$. Then the set of edges in $X \triangle Y \subseteq E(w)$ is $C^{\ell+2}_{\infty\omega}$-definable and hence also the set of neighbours $Z \subseteq V$ of w which are incident with these edges. Assume that Z is a union of blocks from the old partition. Then b would already be definable from the other parameters, a contradiction to the minimality of σ. Hence, the partition can indeed be refined and the new blocks are again $C^{\ell+2}_{\infty\omega}$-definable.

Next we obtain a bound on $\ell = |\sigma \cap \hat{V}^T|$. We know from Equation 1 that

$$\left|S_V / \mathrm{Stab}^{\bullet}_{S_V}(\sigma)\right| \leq 2^{n \cdot q},$$

and also that $\mathrm{Stab}^{\bullet}_{S_V}(\sigma) \leq \mathrm{Stab}_{S_V}(V^*_0) \cap \cdots \cap \mathrm{Stab}_{S_V}(V^*_\ell)$. Moreover, it is easy to see that $\frac{n!}{(n-\ell)!} \leq |S_V / (\mathrm{Stab}_{S_V}(V^*_0) \cap \cdots \cap \mathrm{Stab}_{S_V}(V^*_\ell))|$. It follows that $n!/(n-\ell)! \leq 2^{n \cdot q}$.

By the Stirling formula we can find a constant $c > 0$ such that

$$\ell \leq \frac{q}{(\log n + \log e)} \cdot n - \frac{\log c}{(\log n + \log e)}.$$

Hence, for large enough n it follows that $\ell \leq \epsilon \cdot n$. More generally, the argument shows that the number ℓ of parts in a partition of V as above is, for large n, smaller than $\epsilon \cdot n$. However, to prove our original claim we have to find a support σ of x such that $|\sigma| \leq \epsilon \cdot n$ (by now we only proved that $\ell = |\sigma \cap \hat{V}^T| \leq \epsilon \cdot n$).

Hence, in a second step we consider the elements in σ which are part of $\hat{E}$. Every such element $e^i \in \sigma \cap \hat{E}$ corresponds to an edge $e = \{v, w\}$ in the original complete graph. As above, we can use the e^i to refine the partition of V, because we can define (using e^i) the new block $\{v, w\}$. However, in contrast to the case above, it might happen that the partition cannot be refined although we still have unprocessed $e^i \in \sigma \cap \hat{E}$ left. There are two possible reasons: either we already identified a block $\{v, w\}$ (for instance, by using inner nodes) or we have refined V into one block $\{v\}$ and another block $\{w\}$. We overcome this issue by modifying the strong support σ. In fact, we can replace each $e^i \in \sigma$ where $e = \{v, w\}$ by two inner nodes $a \in v_T^*$ and $b \in w_T^*$ to obtain a new support σ' for x, i.e. $\mathrm{Stab}^\bullet_\Gamma(\sigma') \leq \mathrm{Stab}^\bullet_\Gamma(\sigma) = \mathrm{Stab}(x)$. Note, however, that σ' might no longer be a strong support for x. The question remains how many inner nodes we have to add to cover all edges in the support for which we do not obtain a refinement of our partition. The answer is: not more than we have blocks in our partition after a maximal refinement. This is because the only cases where edges do not lead to refinements of the partition can be resolved by fixing an inner node for one vertex in a block of size one or two. Hence, we can find a new support σ' for x of size $|\sigma'| \leq 2 \cdot \ell$ where ℓ denotes the maximal length of a partition of V that we can get by using first all inner nodes and then some (maximal set of) edge nodes. Since we can choose n large enough such that $\ell \leq \frac{\epsilon}{2} \cdot n$, this suffices to prove our claim. ◀

▶ **Theorem 22** (Dawar, Richerby, Rossman). *Let $\ell \geq 1$, $k \geq 1$ and let $\mathfrak{A}$ and $\mathfrak{B}$ be two $C^{\ell \cdot k}$-homogeneous structures. If $\mathfrak{A} \equiv^C_{\ell \cdot k} \mathfrak{B}$, then $\mathrm{HF}(\mathfrak{A})_\ell \equiv^C_k \mathrm{HF}(\mathfrak{B})_\ell$.*

▶ **Theorem 23.** *Let $\Pi \in CPT$ be a program which activates only strongly supported sets and which has resource bounds $p(n) = n^q$ for some $q \geq 1$. Then Π cannot decide the Cai-Fürer-Immerman query over complete graphs.*

Proof. To obtain a contradiction, assume that there is a CPT-program Π with the above properties. First we translate Π into an equivalent $C^k_{\infty\omega}$-formula φ_Π which simulates Π on $\mathrm{Active}(\Pi, \mathfrak{G}^T)$, i.e. $\mathfrak{G}^T \models \Pi$ if, and only if, $\mathrm{Active}(\Pi, \mathfrak{G}^T) \models \varphi_\Pi$.

Secondly, we choose $m \geq 1$ sufficiently large (according to Lemma 21 for q and $\epsilon = \frac{1}{2k}$) such that all sets $x \in \mathrm{HF}(\mathfrak{G}^T)$ which can be activated by Π in CFI-graphs $\mathfrak{G}^T$ over complete graphs $G = (V, E)$ with $|V| = n \geq m$ vertices have a support of size at most $\frac{1}{2k} \cdot n = \epsilon \cdot n$. Hence $\mathrm{Active}(\Pi, \mathfrak{G}^T) \subseteq \mathrm{HF}(\mathfrak{G}^T)_{\epsilon \cdot n}$. Let $\mathfrak{G}^T$ be an even and $\mathfrak{G}^S$ be an odd CFI-graph over such a complete graph G. It is well-known [6] that $\mathfrak{G}^T \equiv^C_{n/2} \mathfrak{G}^S$. Since $n/2 = \epsilon \cdot n \cdot k$, we can apply Theorem 22 to conclude that $\mathrm{HF}(\mathfrak{G}^T)_{\epsilon \cdot n} \equiv^C_k \mathrm{HF}(\mathfrak{G}^S)_{\epsilon \cdot n}$. Hence, Π cannot distinguish between the two CFI-graphs. ◀

This result provides a deeper understanding of the expressive power of fragments of Choiceless Polynomial Time. Interpretation Logic, which was introduced in [12], characterises CPT and some of its fragments in terms of first-order interpretations (with the Härtig quantifier for equicardinality testing). This framework allows to iterate a first-order interpretation until a first-order halting condition is satisfied, and evaluates a first-order sentence on the resulting structure (for details, see [12]), with polynomial bounds on the number of iterations and the size of the intermediate structures. The fragment of interpretation logic that arises when only interpretations without congruences are allowed, i.e. different tuples cannot be identified to obtain a single element of the interpreted structure, has already been shown to be strictly

weaker than CPT. Without counting (respectively equicardinality quantifiers), this fragment corresponds to the database query language $\text{while}_{\text{new}}$, which permits construction of new elements for definable *tuples*. For the fragment with counting, it could be shown [12] that, on structures with bounded colour class size, only CPT computations with sets of bounded rank can be simulated. Theorem 23 now implies that the fragment not only fails to define sets of unbounded rank, but also any kind of set-like objects.

▶ **Corollary 24.** *Polynomial-Time Interpretation Logic without congruences cannot define the CFI query over complete graphs.*

Proof. A CPT-program simulating an Interpretation Logic computation constructs exactly those sets that represent the interpreted structures, i.e. a polynomially sized, first-order definable set of *tuples* for each structure, and relations in the structures. These objects are strongly supported. ◀

8 Conclusion and Future Work

We generalise previous expressibility results for the CFI query in Choiceless Polynomial Time to graphs with colour classes of logarithmic size and graph classes where the maximal degree is linear. In the latter case, CPT over sets of bounded rank suffices, which illustrates that on large enough input structures, the full power of CPT is not exhausted even for structures that, like the CFI-graphs, have many symmetries. The first result suggests that results for structures with bounded colour class size may be lifted to larger colour classes.

Clearly, a CPT procedure for the CFI query over arbitrary graphs remains desirable. Especially since graph classes with vertices of large (linear) degree are already covered, graphs with bounded degree, especially the possibly simpler case of 3-regular graphs, seem to be an important benchmark for the general case.

Another promising direction for future work seems to be the connection between CFI-graphs and linear algebra. The CFI problems studied here may give rise to classes of linear equation systems that are solvable in CPT, which would advance the study of expressibility of linear algebra in CPT and thus the connection between CPT and Rank Logic.

On the other hand, our inexpressibility result for CPT without set-like objects separates two of the few known natural fragments of CPT. Our characterisation of set-like objects could provide new insights for other sequence-based formalisms.

References

1 F. Abu Zaid, E. Grädel, M. Grohe, and W. Pakusa. Choiceless Polynomial Time on structures with small Abelian colour classes. In *MFCS 2014*, volume 8634 of *LNCS*, pages 50–62. Springer, 2014.

2 A. Blass. Why sets? In *Pillars of Computer Science*, volume 4800 of *LNCS*, pages 179–198. Springer, 2008. doi:10.1007/978-3-540-78127-1_11.

3 A. Blass, Y. Gurevich, and S. Shelah. Choiceless polynomial time. *Annals of Pure and Applied Logic*, 100(1):141–187, 1999.

4 A. Blass, Y. Gurevich, and S. Shelah. On polynomial time computation over unordered structures. *J. Symb. Logic*, 67(3):1093–1125, 2002.

5 Andreas Blass, Yuri Gurevich, and Jan Van den Bussche. Abstract state machines and computationally complete query languages. In *International Workshop on Abstract State Machines*, pages 22–33. Springer, 2000.

6 J. Cai, M. Fürer, and N. Immerman. An optimal lower bound on the number of variables for graph identification. *Combinatorica*, 12(4):389–410, 1992.

7 A. Chandra and D. Harel. Structure and complexity for relational queries. *Journal of Computer and System Sciences*, 25:99–128, 1982.

8 A. Dawar. The nature and power of fixed-point logic with counting. *ACM SIGLOG News*, pages 8–21, 2015.

9 A. Dawar, M. Grohe, B. Holm, and B. Laubner. Logics with rank operators. In *LICS 2009*, pages 113–122, 2009.

10 A. Dawar, D. Richerby, and B. Rossman. Choiceless polynomial time, counting and the Cai–Fürer–Immerman graphs. *Ann. Pure Appl. Logic*, 152(1), 2008.

11 E. Grädel and M. Grohe. Is Polynomial Time Choiceless? In *Fields of Logic and Computation II.*, volume 9300 of *LNCS*, pages 193–209. Springer, 2015.

12 E. Grädel, Ł. Kaiser, W. Pakusa, and S. Schalthöfer. Characterising Choiceless Polynomial Time with First-Order Interpretations. In *LICS*, 2015.

13 E. Grädel and W. Pakusa. Rank logic is dead, long live rank logic! *CoRR (a conference version appeared in the proceedings of CSL'15)*, abs/1503.05423, 2015.

14 M. Grohe. The quest for a logic capturing PTIME. In *Logic in Computer Science, 2008, (LICS'08)*, pages 267–271. IEEE, 2008.

15 Y. Gurevich. Logic and the challenge of computer science. In *Current Trends in Theoretical Computer Science*. Computer Science Press, 1988.

16 N. Immerman. Relational queries computable in polynomial time. *Inf. and Control*, 68:86–104, 1986.

17 B. Laubner. *The Structure of Graphs and New Logics for the Characterization of Polynomial Time*. PhD thesis, HU Berlin, 2011.

18 B. Rossman. Choiceless computation and symmetry. In *Fields of Logic and Computation*, LNCS, pages 565–580. Springer, 2010.

19 M. Y. Vardi. The complexity of relational query languages. In *STOC'82*, pages 137–146. ACM Press, 1982.