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

WIED PAKUSA, SVENJA SCHALTHÖFER, Mathematical Foundations of Computer Science, RWTH Aachen University
ERKAL SELMAN, Logic and Theory of Discrete Systems, RWTH Aachen University

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, because of isomorphism-invariance, it is choiceless in the sense that it cannot select an arbitrary element of a set—an operation that is crucial for many classical algorithms. CPT can define many interesting Ptime queries, including (a certain 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. For the case where the CFI-graphs arise from ordered graphs, Dawar, Richerby, and Rossman proved that the CFI-query is CPT-definable. However, definability of the CFI-query over general graphs remains open.

Our first contribution generalises the result by Dawar, Richerby, and Rossman to the variant of the CFI-query derived from graphs with colour classes of logarithmic size, instead of colour class size one. Second, we consider the CFI-query over graph classes where the maximal degree is linear in the size of the graphs. For the latter, 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, then no such procedure would be definable. We introduce a notion of "sequencelike objects" based on the structure of the graphs' symmetry groups, and we show that no CPT-program that only uses sequencelike 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.

CCS Concepts: • **Theory of computation → Finite Model Theory**;

Additional Key Words and Phrases: Cai-Fürer-Immerman, choiceless polynomial time, descriptive complexity, finite model theory, logic for PTIME

Authors' addresses: W. Pakusa, Department of Computer Science, University of Oxford, Wolfson Building, Parks Road, Oxford, OX1 3QD, UK; email: wied.pakusa@cs.ox.ac.uk; S. Schalthöfer, Mathematische Grundlagen der Informatik, RWTH Aachen, 52056 Aachen, Germany; email: schalthoefer@logic.rwth-aachen.de; E. Selman, Lehrstuhl für Informatik 7, RWTH Aachen, 52056 Aachen, Germany; email: selman@informatik.rwth-aachen.de.

https://doi.org/10.1145/3154456

ACM Reference format:
Wied Pakusa, Svenja Schalthöfer, and Erkal Selman. 2018. Definability of Cai-Fürer-Immerman Problems in Choiceless Polynomial Time. *ACM Trans. Comput. Logic* 19, 2, Article 7 (February 2018), 27 pages.
https://doi.org/10.1145/3154456

1 INTRODUCTION

One of the most important questions in descriptive complexity theory is whether there is a logic capturing Ptime (Chandra and Harel 1982; Grohe 2008; Gurevich 1988).

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 (Immerman 1986; Vardi 1982). 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, that 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 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 Dawar (2015).

However, Cai et al. (1992) found a very clever construction that 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 et al. (2009). It is still open whether certain variants of Rank Logic (Grädel and Pakusa 2015) 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 (1988), of what would constitute a logic for Ptime. Choiceless Polynomial Time, introduced by Blass et al. (1999), 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. Due to isomorphism-invariance, Choiceless Polynomial Time, unlike conventional machine 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. Whereas other 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 (Blass et al. 2002; Laubner 2011). 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 CFI-graphs are preordered with colour classes of size two (Dawar et al. 2008).
- —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 (Abu Zaid et al. 2014).

For a more detailed survey of some recent insights, see also Grädel and Grohe (2015).

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 rather simple classes 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 special case of the graph isomorphism problem. With each *connected* graph $G = (V, E)$, the CFI-construction associates a set of CFI-graphs $\{\mathfrak{G}^T : T \subseteq V\}$ over G, 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 (Cai et al. 1992). Furthermore, it is definable in Rank Logic (Dawar et al. 2009) and in Choiceless Polynomial Time without counting in case the underlying graph G is *linearly ordered*, but not definable in CPT using only sets of bounded rank (Dawar et al. 2008).

The result of Cai, Fürer, and Immerman works particularly when applying 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 (depending on the exact construction, which may differ in technical details while admitting the same results). 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 et al. (2002), 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 led 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 et al. (2008) were able to confirm this later for the special case where the underlying graphs are linearly ordered (as it is the case in the construction used by Blass et al. (2002) in their expressibility result for padded CFI graphs). 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 that 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 article, we set out to identify new classes of graphs over which the CFI-query is CPT-definable. Moreover, we analyse the resources that are required by CPT-programs to decide the CFI-query. A short version of this article appeared as Pakusa et al. (2016).

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 total preorder $\preceq$ on the universe, which may contain classes of elements where $\preceq$ is symmetric, 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 linearly 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 that 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.

Second, we strengthen a claim from Dawar et al. (2008) 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 $\frac{n}{k}$, then the associated CFI-graph is of size at least $2^{\frac{n}{k}}$. 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 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 (2000): setlike data structures are more powerful than sequencelike data structures (see also (Blass 2008).) We generalise this result to the case of counting and show that CPT-programs

that decide the CFI-query over complete graphs have to use setlike objects. In particular, this yields an interesting lower bound for a fragment of Interpretation Logic, a characterisation of Choiceless Polynomial Time, which was presented by Grädel et al. (2015): 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 (2010). Let τ be a relational signature and let $\mathfrak{A} = (A, \tau^{\mathfrak{A}})$ 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*, i.e., objects 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

The semantics of the above comprehension term is $\{s^{\mathfrak{A}}(\bar{a}, b) : y \in r^{\mathfrak{A}}(\bar{a}) : \mathfrak{A} \models \varphi(\bar{a}, b)\}$ 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

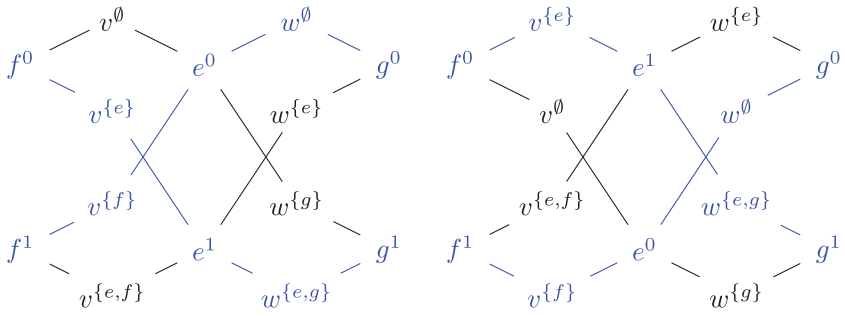

Fig. 1. A subgraph of the graph $\mathfrak{G}$. The subgraph highlighted on the left corresponds to the CFI-graph $\mathfrak{G}^{\{v\}}$. The automorphism ρ_e maps $\mathfrak{G}^{\{v\}}$ to $\mathfrak{G}^{\{w\}}$, which is highlighted on the right.

the Kuratowski encoding of pairs, so this is indeed only a change of syntax. We denote tuples as objects $\langle a_1, \ldots, a_k \rangle$.

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 $V', V'' \subseteq V$, then $E[V']$ 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 $w \in V''$. We also write (V', v) to denote the cut $(V', \{v\})$.

A preordered graph $G^{\preceq} = (V, E, \preceq)$ is a graph $G = (V, E)$ equipped with a total preorder $\preceq$ on the vertices. When working with a fixed ordered version $G^{\preceq}$ of G, we may also refer to $G^{\preceq}$ as G. A set where $\preceq$ is symmetric is called a *colour class*. We write C_i for the ith colour class in the linear order that $\preceq$ 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 remainder of this section, 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 $\{\rho_e := \rho_{\{e\}} : e \in E\}$.

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 (1992), 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 C is the class of all $\mathfrak{G}^T$ for $G \in C$. If G is preordered by $\preceq$, then $\preceq$ induces an order on $\mathfrak{G}$ (and thus 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 et al. (2008) constructs an object that 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 $\mathfrak{G}$.

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 $\mu_{|V|}$, the set $\mu_{|V|}$ itself remains 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 Dawar et al. (2008) 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 5.1. *Let $\mathcal{K}$ be the class of connected, preordered graphs $G = (V, E, \preceq)$ 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 $\mu_i, \tilde{\mu}_i$ described above that represent the parity of $|T \cap \{v_1, \ldots, v_i\}|$. 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 $\mu_M, \tilde{\mu}_M$ 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. Here, $\uplus$ denotes that the sets N and $\{v\}$ are already disjoint.

To replace the e^i with their labels from $\{0, 1\}$, the algorithm by Dawar et al. (2008) 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 $\mu_M, \tilde{\mu}_M$. 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 $\nu_M, \tilde{\nu}_M$ 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 $\tau_v, \tilde{\tau}_v$ for $v \in V$ are defined as by Dawar et al. (2008), with one difference: To allow for assigning labels 0 or 1 to several vertices at once 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 5.2. Let $v \in V$. Then,

$$\tau_v^T = \left\{\langle N(v^X), 0\rangle : v^X \in v_T^*\right\} \text{ and}$$
$$\tilde{\tau}_v^T = \left\{\langle N(v^X), 0\rangle : v^X \in v^* \setminus v_T^*\right\},$$

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

Definition 5.3. 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 5.4. Let $T \subseteq V$, $v \in V$, $M \in \mathcal{M}$ and $v \notin M$. Then,

$$\mu_{\{v\}}^T = \tau_v^T,$$
$$\tilde{\mu}_{\{v\}}^T = \tilde{\tau}_v^T,$$
$$\mu_{M,v}^T = \left\{\left\langle \mu_M^T, \tau_v^T\right\rangle, \left\langle \tilde{\mu}_M^T, \tilde{\tau}_v^T\right\rangle\right\},$$
$$\tilde{\mu}_{M,v}^T = \left\{\left\langle \mu_M^T, \tilde{\tau}_v^T\right\rangle, \left\langle \tilde{\mu}_M^T, \tau_v^T\right\rangle\right\},$$

and, for $|M| > 1$,

$$\mu_M^T = \left\{\mu_{N,w}^T : N \in \mathcal{M} \text{ and } N \uplus \{w\} = M\right\} \text{ and}$$
$$\tilde{\mu}_M^T = \left\{\tilde{\mu}_{N,w}^T : N \in \mathcal{M} \text{ and } N \uplus \{w\} = M\right\}.$$

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

Lemma 5.5. $\mu_M^T = \mu_M^S \Leftrightarrow \tilde{\mu}_M^T = \tilde{\mu}_M^S \Leftrightarrow |S \cap M| \equiv |T \cap M| \pmod 2$ *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 $\nu_M, \tilde{\nu}_M$ 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 5.6. 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 Dawar et al. (2008) 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 5.7. *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 $\nu_M, \tilde{\nu}_M$ can be defined. The sets $\nu, \tilde{\nu}$ 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 5.8. Let $M \in \mathcal{M}$ with $|M| > 1$. Then,

$$\nu_M = \left\{B_{v,N}(\nu_{(N,v)}) : N \uplus \{v\} = M \text{ and } N \in \mathcal{M}\right\} \text{ and}$$
$$\tilde{\nu}_M = \left\{B_{v,N}(\tilde{\nu}_{(N,v)}) : N \uplus \{v\} = M \text{ and } N \in \mathcal{M}\right\}.$$

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 $\mu_M^T, \tilde{\mu}_M^T$, the sets $\nu_M^T, \tilde{\nu}_M^T$ characterise the parity of $|T \cap M|$. To show this, and make the $\nu_M^T, \tilde{\nu}_M^T$ more accessible to further analysis, we show that ν_M (respectively, $\tilde{\nu}_M$) arises from μ_M (respectively, $\tilde{\mu}_M$) 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 5.9. 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 5.10. *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. For $|M| = 1$, the statements are trivial. So, let $|M| > 1$. (1) follows directly from the induction hypothesis for (4), distinguishing the cases $v \in S \triangle T$ and $v \notin S \triangle T$.

To prove (2), let $e \in (N, \{v\})$. So, $e = \{v, w\}$ for some $w \in N$. ρ_e maps $\mathfrak{G}^T$ to $\mathfrak{G}^S$ such that $S \triangle T = \{v, w\}$. Then, (2) follows from (1), because $|S \cap M| \equiv |T \cap M|$.

By (2), the expression $B_{v,N}(\nu_{(N,v)})$ is well-defined. In particular, $B_{v,N}(\nu_{(N,v)}) = B_{(N,\{v\})}(\nu_{(N,v)})$. Moreover, by the induction hypothesis and because no edge gadget for an edge in $E[N]$ occurs in τ_v or $\tilde{\tau}_v$, it follows that $\nu_{(N,v)} = B_{E[N]}(\mu_{N,v})$. But then $\nu_M = B_{(N,\{v\})}(B_{E[N]}(\mu_M)) = B_{E[M]}(\mu_M)$. So, (3) is true.

To obtain (4), it suffices to show that $B_{v,M}(\nu^T_{(N,v)}) = B_{v,M}(\nu^S_{(N,v)})$ if, and only if, $\nu^T_{(N,v)} = \nu^S_{(N,v)}$, since $\nu^T_{(N,v)}$ characterises the parity of $|T \cap M|$. By (3), this follows if $B_E(\mu^T_{N,v}) = B_E(\mu^S_{N,v})$ implies that $\mu^T_{N,v} = \mu^S_{N,v}$, which is an easy induction on $|M|$. □

Corollary 5.11. *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 5.12.

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

Lemma 5.13. *$p(\nu^T_V) = 0$ if, and only if, $|T|$ is even.*

Proof. As, by Lemma 5.10, $\nu^T_V = B_E(\mu^T_V)$ 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, we show the following statements for all $M \subseteq V$ and $v \in V \setminus M$ by induction on $|M|$, where $P(x)$ denotes $p(B_E(x))$:

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

First, let $T = \emptyset$. Each $\tau_v^\emptyset$ consists of neighbourhoods of the vertices v^X, which contain an even number of vertices of the form e^1. So, all parity bits are 0 in $B_E(\tau_v^\emptyset)$, and 1 in $B_E(\tilde{\tau}_v^\emptyset)$. So, $P(\mu_{\{v\}}^\emptyset) = 0$ and $P(\tilde{\mu}_{\{v\}}^\emptyset) = 1$, and thus,

$$P\left(\mu_{M,v}^\emptyset\right) = \left(B_E\left(\mu_M^\emptyset\right) + B_E\left(\tau_v^\emptyset\right)\right) \cdot \left(B_E\left(\tilde{\mu}_M^\emptyset\right) + B_E\left(\tilde{\tau}_v^\emptyset\right)\right) = 0.$$

Since $P(\mu_{N,v}^\emptyset) = 0$ for all sets N and vertices v with $N \uplus \{v\} = M$, it follows that $P(y) = 0$ for all $y \in \mu_M$. Thus, $P(\mu_M^\emptyset) = 0$.

If $T = \{x\}$, then, by definition of τ_v and $\tilde{\tau}_v$,

$$P\left(\mu_{\{v\}}^T\right) = 1 \Leftrightarrow P\left(\tilde{\mu}_{\{v\}}^T\right) = 0 \Leftrightarrow v = x.$$

Now if $x \in M \uplus \{v\}$, then either $x \in M$ and $P(\mu_M^{\{x\}}) = 1$ by the induction hypothesis, or $v = x$ and $P(\tau_v^{\{x\}}) = 1$. In both cases, $P(\mu_{M,v}^{\{x\}}) = 1$. Finally, since again the value of $\mu_{M,v}^{\{x\}}$ only depends on $M \uplus \{v\}$,

$$p\left(B_E\left(\mu_M^{\{x\}}\right)\right) = 1 \Leftrightarrow p\left(B_E\left(\tilde{\mu}_M^{\{x\}}\right)\right) = 0 \Leftrightarrow x \in M. \qquad \square$$

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

Proof. For $k \leq |V| =: n$, let $\mu_k = \{\mu_M, \tilde{\mu}_M : M \in \mathcal{M} \text{ and } |M| = k\}$. We show by induction on k that the number of elements in $\operatorname{tc}(\mu_k) \setminus \operatorname{tc}(\mu_{k-1})$ is polynomial in n. The lemma then follows, because $k \leq n$.

For $k = 1$, consider all $\tau_v, \tilde{\tau}_v$ for $v \in V$. Each τ_v or $\tilde{\tau}_v$ is a set of neighbourhoods of the vertices v^X. So, the transitive closure contains $\leq |\mathfrak{G}|$ many neighbourhoods consisting of vertices in $\hat{E}(v)$.

For $k > 1$, 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 $\mu_{N,v}$ and $\tilde{\mu}_{N,v}$ for sets N with $|N| = k - 1$. $\square$

By Lemma 5.10, $|\operatorname{tc}(\nu_M)| \leq |\operatorname{tc}(\mu_M)|$ for all $M \subseteq V$. So, our construction is CPT-definable, which completes the proof of Theorem 5.1.

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 Dawar et al. (2008).

Theorem 6.1. *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 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 by Dawar et al. (2008), 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 $\tilde{\tau}_v$. Working with subsets of V allows to circumvent the nesting of sets that is noticeable in Definition 5.4, thus keeping the rank of the sets used here small.

Definition 6.2. Let $v \in V$ and $M \subseteq V$ with $|M| \geq 2$. Then, $\mu^T_{\{v\}} = \{\{\tau_v\}\}$ and $\tilde{\mu}^T_{\{v\}} = \{\{\tilde{\tau}_v\}\}$. For larger M, let $\bar{\mu}^T_M = \{\{\bar{\tau}^T_v : v \in M\} : \bar{\tau}^T_v \in \{\tau^T_v, \tilde{\tau}^T_v\}\}$. Then,

$$\begin{aligned}\mu^T_M =& \{m \in \bar{\mu}^T_M : \text{ the number of } \tilde{\tau}^T_v \text{ in } m \text{ is even}\}, \text{ and}\\ \tilde{\mu}^T_M =& \{m \in \bar{\mu}^T_M : \text{ the number of } \tilde{\tau}^T_v \text{ in } m \text{ is odd}\}.\end{aligned}$$

The sets $\mu_M, \tilde{\mu}_M$ for $M \subseteq V$ defined in this way again characterise the parity of $|T \cap M|$.

Lemma 6.3. $\mu^T_M = \mu^S_M \Leftrightarrow \tilde{\mu}^T_M = \tilde{\mu}^S_M \Leftrightarrow |S \cap M| \equiv |T \cap M| \pmod 2$ *and*
$\mu^T_M = \tilde{\mu}^S_M \Leftrightarrow \tilde{\mu}^T_M = \mu^S_M \Leftrightarrow |S \cap M| \not\equiv |T \cap M| \pmod 2$.

Proof. The first equivalence of both statements follows directly by induction on $|M|$.

To see that $\mu^T_M = \mu^S_M$ implies $|S \cap M| \equiv |T \cap M|$, consider the set $\{\tau^T_v : v \in M\} \in \mu^T_M$. If that set also occurs as $m \in \mu^S_M$, then the number of $\tilde{\tau}^S_v$ in m is even. So, there is an even number of vertices with $\tau^T_v = \tilde{\tau}^S_v$ and thus $|S \triangle T|$ is even. With the same argument, if $\tilde{\mu}^T_M = \mu^S_M$, then $|S \triangle T|$ is odd.

The other direction is another simple induction on $|M|$. □

We again use the mappings $B_{v,M}$ arising from Lemma 5.7 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 $\mu_M, \tilde{\mu}_M$ 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 $\tilde{\mu}_N$. 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.

Definition 6.4. Let $M, T \subseteq V$ and $v \in V$.

$$\begin{aligned}\nu^T_{\{v\}} &= \left\{\left\{\tau^T_v\right\}\right\}, \tilde{\nu}^T_{\{v\}} = \left\{\left\{\tilde{\tau}^T_v\right\}\right\},\\ \nu^T_{(M,v)} &= \left\{X \cup \left\{\tau^T_v\right\} : X \in \nu^T_M\right\} \cup \left\{X \cup \left\{\tilde{\tau}^T_v\right\} : X \in \tilde{\nu}^T_M\right\},\\ \tilde{\nu}^T_{(M,v)} &= \left\{X \cup \left\{\tilde{\tau}^T_v\right\} : X \in \nu^T_M\right\} \cup \left\{X \cup \left\{\tau^T_v\right\} : X \in \tilde{\nu}^T_M\right\},\end{aligned}$$

and $\nu^T_M = B_{w,N}(\nu^T_{(N,w)})$ and $\tilde{\nu}^T_M = B_{w,N}(\tilde{\nu}^T_{(N,w)})$ for some (all) $N \subseteq V, w \in V$ such that $N \uplus \{w\} = M$.

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

Analogously to Lemma 5.10, we can show that the sets $\nu_M, \tilde{\nu}_M$ are well-defined, again using the mappings B_F from Definition 5.9, 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.

Definition 6.5.

$$p(x) = \begin{cases} i & x = \left\langle N\left(v^X\right), i\right\rangle, \\ \sum_{y \in x} p(y) & x \in \nu_V, \\ \prod_{y \in x} p(y) & x = \nu_V \text{ or } x = \tau_v \text{ for some } v \in V, \end{cases}$$

where the sums and products are computed in $\mathbb{F}_2$.

Lemma 6.6. $p(\nu_V^T) = |T| \mod 2$.

Proof. By the analogue of Lemma 5.10, $\nu_V^T = B_E(\mu_V^T)$. First, consider a set τ_v^T (the argument for $\tilde{\tau}_v^T$ is analogous). For each neighbourhood in τ_v^T, the parity bit is 1 if, and only if, $v \in T$. Since all parity bits in τ_v^T coincide, the product is also 1 if, and only if, $v \in T$.

Let $m \in \mu_V^T$. Then, the number of $\tilde{\tau}_v^T$ in m is even. So, $p(B_E(m)) = 1$ if, and only if, m contains an odd number of τ_v^T with $p(B_E(\tau_v^T)) = 1$, which holds if, and only if, $|T|$ is odd. All $m \in \mu_V^T$ get the same value, so $p(B_E(\mu_V^T)) = p(\nu_V^T) = 1$ if, and only if, $|T|$ is odd. □

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 bounded rank. Furthermore, we need to show that the size of the transitive closure of each ν_M is polynomial in $|\mathfrak{G}|$.

Lemma 6.7. *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 6.8. *If G has a vertex of degree $\geq \frac{|V|}{k}$, then 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 that decides the Cai-Fürer-Immerman query over ordered complete graphs has to use setlike objects. In fact, observe that the CPT-procedure that we presented in the previous section strongly makes use of the computational power of sets in order to go through all possible subsets of the vertex set of the underlying complete graph. The simple observation was that if this underlying complete graph has n vertices, then the corresponding CFI-graph has about $n \cdot 2^n$ elements, which means that activating all 2^n subsets of the vertex set does not violate the polynomial resource bounds of our CPT-program. On the other hand, if we had to work with tuples only, then it would be unclear how to represent these 2^n subsets in a succinct way. Indeed, just enumerating them in any order would lead to about $n!$

different objects, which clearly is super-polynomial in the size of the CFI-graph. As we will see this blow-up cannot be avoided and it is impossible to decide the CFI-query with a restriction of CPT to tuples only.

Before we can start, we first have to find a reasonable generalisation of tuples to formalise the notion of *setlike* and *tuplelike* objects. To motivate our definition, let us come back to our example from above. Let $\mathcal{K}_n = (V, E)$ be a complete graph over $|V| = n$ vertices. The automorphism group of $\mathcal{K}_n$ 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 $\mathcal{K}_n$ is invariant under the action of Γ, every CPT-program that constructs x also has to construct the whole orbit of x, that is $\mathrm{Orbit}(x) = \{\pi(x) : \pi \in \Gamma\}$. Of course, the same holds for every program that constructs 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 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 $\mathrm{Stab}_\Gamma(x)$ of x is much larger than the size of the stabiliser $\mathrm{Stab}_\Gamma(y)$ of y. Indeed, $|\mathrm{Stab}_\Gamma(x)| = k! \cdot (n-k)!$ and $|\mathrm{Stab}_\Gamma(y)| = (n-k)!$. Obviously, the reason for this is that 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 $\mathrm{Stab}^\bullet_\Gamma\{a_1, \dots, a_k\} = \{\pi : \pi(a_i) = a_i \text{ for all } i < k\}$ of $a_0, \dots, a_{k-1}$ coincides with $\mathrm{Stab}_\Gamma(y)$ while it is a subgroup of $\mathrm{Stab}_\Gamma(x)$ of index $k!$.

Definition 7.1. 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 $\mathrm{Stab}^\bullet_\Gamma(\sigma) \leq \mathrm{Stab}_\Gamma(x)$, and σ is called a *strong support* of x if $\mathrm{Stab}^\bullet_\Gamma(\sigma) = \mathrm{Stab}_\Gamma(x)$.

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 $\mathrm{Stab}_\Gamma(x) = \Gamma$ 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 that are "not setlike," or more intuitively, that are "sequencelike." Those objects enforce an inherent order on the supporting atoms. We aim to prove Theorem 7.25: no CPT-program that can only access strongly supported objects can express the CFI-query over ordered complete graphs (the same result for unordered complete graphs follows immediately).

Our proof is based on the techniques developed by Dawar et al. (2008). It is well-known that every CPT-program Π can be translated into a formula φ_Π of infinitary logic with counting and m variables, for a certain m that depends on Π (we denote this logic by $\mathbf{C}^m$), 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 that are *activated* (intuitively: have to be constructed) during the run of Π on $\mathfrak{A}$. For details see Dawar et al. (2008), Blass et al. (1999), and Blass et al. (2002). 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 $\mathbf{C}^m$-formula ($\mathrm{Active}(\Pi, \mathfrak{A}) \equiv^C_m \mathrm{Active}(\Pi, \mathfrak{B})$), 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 $\mathrm{Active}(\Pi, \mathfrak{A}) \equiv^C_m \mathrm{Active}(\Pi, \mathfrak{B})$ is combinatorially challenging, because these structures contain highly nested sets over the two input structures.

— The structures Active($\Pi, \mathfrak{A}$) and Active($\Pi, \mathfrak{B}$) depend on the program Π. Since we want to obtain an undefinability result, for all we know this program Π can behave completely arbitrarily.

There is a nice approach, which was first used by Blass et al. (1999) and which was later refined and generalised by Dawar et al. (2008, 2010), which solves both problems at once. Very roughly, the idea is that for certain structures $\mathfrak{A}$ we can over-approximate Active($\Pi, \mathfrak{A}$) by a substructure of HF($\mathfrak{A}$) that contains only objects with sufficiently small supports.

More importantly, if the orbits of tuples in the structure $\mathfrak{A}$ are definable in $\mathbf{C}^m$ for a certain m ($\mathfrak{A}$ is then called $\mathbf{C}^m$-homogeneous), then we can obtain a description of the (highly nested) 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 of dealing with highly nested sets in HF($\mathfrak{A}$) to flat objects over $\mathfrak{A}$.

$\mathbf{C}^m$-equivalence of the substructures of HF($\mathfrak{A}$) and HF($\mathfrak{B}$) is shown by giving a winning strategy for Duplicator in the bijective pebble game. Using the description of sets in terms of their supports, this strategy can be obtained from a winning strategy on the CFI-graphs themselves where Spoiler places his pebbles on the supports.

Our aim is to adapt the approach of Dawar, Richerby, and Rossman for our application. To this end CFI-graphs over ordered complete graphs should satisfy the following requirements:

— Every strongly supported set activated by a CPT-program has a strong support that is in a normal form and sufficiently small (Lemma 7.3).
— Duplicator wins the bijective pebble game where Spoiler can put pebbles on a constant number of strong supports (Lemma 7.10). This is not evident from known results, because our strong supports do not have constant size.
— The graphs are homogeneous with respect to tuples that can occur as strong supports (Lemma 7.12).

These properties will then imply that the winning strategy on the CFI-graphs can be translated to a winning strategy on the strongly supported hereditarily finite sets (Lemma 7.20).

In the following, we denote by $\mathcal{K}_n^<$ the complete graph of size n with a linear order on the vertices. Note that we always assume the CFI-graphs over $\mathcal{K}_n^<$ to be equipped with the total preorder induced by the order on $\mathcal{K}_n^<$.

First, we formalise the normal form for strong supports. Since any vertex in a vertex gadget is uniquely determined by a set of adjacent vertices from edge gadgets, it suffices to consider strong supports that only contain edge gadgets. However, a strong support can then contain $n - 2$ edge gadgets instead of a single vertex gadget.

Definition 7.2. Fix $k, n \in \mathbb{N}$.

— A (k-sized) *molecule* is either the empty tuple, the singleton a for any atom a, or a tuple $\alpha = (a_1, \ldots, a_s)$ of atoms such that $s \leq nk$ and each a_i is an element of an edge gadget.
— For a CFI-graph $\mathfrak{A}$ over $\mathcal{K}_n^<$, we denote by $\mathrm{HF}(\mathfrak{A})_{k-\mathrm{ss}}$ the substructure of HF($\mathfrak{A}$) consisting of all objects whose transitive closure only contains objects with a strong support that is a molecule.

When referencing molecules, we assume the parameter k to be fixed. Note that our definition of molecules differs from that from Dawar et al. (2008), where the length of molecules is bounded by a constant.

We first show that every strongly supported set occurring in the run of a CPT-program has a strong support that is a molecule for some k that depends only on the program. Then, every object

that is *transitively strongly supported*, i.e., every strongly supported object whose transitive closure contains only objects that are in turn strongly supported, is in the domain of $\mathrm{HF}(\mathfrak{A})_{k-\mathrm{ss}}$.

Lemma 7.3 (Size of Strong Supports). *Let $\bar{\Pi} = (\Pi, n^q)$ be a* CPT*-program. Every strongly supported set activated in the run of $\bar{\Pi}$ on a CFI-graph $\mathfrak{G}^T$ over $\mathcal{K}_n^<$ has a strong support that consists of $< 4nq$ vertices that are all in edge gadgets.*

In the following, we fix $k = 4q$.

Note that any strong support can be transformed into a strong support consisting only of edge vertices by replacing every vertex v^X by its neighbouring edge gadgets. Furthermore, whenever there is a strong support containing both vertices from the same edge gadget, then the sequence where any one of the two vertices is removed is still a strong support.

To show that the strong supports cannot grow too large, we show that the orbit of every set with a large strong support is already too large to be activated by the CPT-program.

By the nature of strong supports, it actually suffices to show a lower bound on the orbit of the strong support:

Lemma 7.4. *Let x be strongly supported by α. Then, $|Orbit(x)| = |Orbit(\alpha)|$.*

Proof. Immediate from the orbit-stabiliser theorem. □

So, Lemma 7.3 follows from

Lemma 7.5. *Let $\mathfrak{G}^T$ be a CFI-graph over $\mathcal{K}_n^<$, and let α be a tuple of vertices from $> nk$ edge gadgets. Then, $|Orbit(\alpha)| > 2^{\frac{nk}{4}}$.*

This lemma is shown by constructing a set of $\frac{nk}{4}$ automorphisms, every one of which flips an edge that occurs in α but is fixed by all other automorphisms in the set. Combining these automorphisms then yields $2^{\frac{nk}{4}}$ distinct tuples in $\mathrm{Orbit}(\alpha)$. Such automorphisms are induced by sets of edge sets called large automorphism sets.

Definition 7.6. Let $\mathcal{K}_n^< = (V, E, <)$. The set $\mathcal{F} \subseteq \mathcal{P}(E)$ is a *large automorphism set* with respect to $A \subseteq E$ if every set in $\mathcal{F}$ is a cycle and there is a set $D \subseteq A$ of size $\geq \frac{nk - \frac{n}{2}}{2}$ such that, for every edge $d \in D$, there is exactly one $F \in \mathcal{F}$ such that $d \in F$ and $d' \notin F$ for every $d \neq d' \in D$.

D is a sufficiently large set of edges that are flipped by exactly one of the automorphisms induced by the sets in $\mathcal{F}$.

For any edge $\{u, v\}$ occurring in α that is incident to edges $\{v, w\}$, $\{u, w\}$ that do not occur in α, the automorphism ρ_F for $F = \{\{u, v\}, \{v, w\}, \{u, w\}\}$ flips $\{u, v\}$ and fixes all other edges occurring in α. If such edges do not exist, i.e., if for every vertex w, either $\{v, w\}$ or $\{u, w\}$ occurs in α, then we call $\{u, v\}$ blocked. Otherwise, $\{u, v\}$ is non-blocked.

To show that a large automorphism set exists, we give an algorithm to compute it. It first constructs as many automorphisms as possible from every blocked edge that has not been processed yet, and then constructs the automorphisms described above for the non-blocked edges.

Lemma 7.7. *On $\mathcal{K}_n^<$, Algorithm 1 traverses the while-loop at most $\frac{n}{2}$ times.*

Proof. Let $e = \{u, v\}$ be the edge chosen in the first line of the loop. After the loop, all edges incident to u and all edges incident to v are in $\bigcup \mathcal{F}$. Furthermore, neither u nor v had that property before the loop was entered, because at this point, e was not in $\bigcup \mathcal{F}$. So, after $\frac{n}{2}$ runs of the loop, all n vertices have the property. This means that there is no edge left in $A \setminus \bigcup \mathcal{F}$. □

Lemma 7.8. *The set $\mathcal{F}$ computed by Algorithm 1 is a large automorphism set with respect to A.*

ALGORITHM 1: Computing Large Automorphism Sets

Input: A complete graph $\mathcal{K}_n^< = (V, E, <)$ and a set $A \subseteq E$
Output: A set $\mathcal{F}$ that is a large automorphism set with respect to A

```
1  while there is a blocked edge in A \ ⋃F do
2  |  choose a blocked edge e ∈ A \ ⋃F;
3  |  foreach triangle {e, f, g} containing e do
4  |  |  if f, g ∉ ⋃F then
5  |  |  |  F := F ∪ {{e, f, g}};
6  |  |  end
7  |  end
8  end
9  foreach edge e = {u, v} ∈ A \ ⋃F do
10 |  choose a vertex w such that {v, w} ∉ A and {u, w} ∉ A;
11 |  F := F ∪ {{e, {u, w}, {v, w}};
12 end
```

Proof. Every edge set added to $\mathcal{F}$ by the algorithm is a cycle. So, it suffices to show that there exists a set D as required by the definition.

Whenever an edge other than the blocked edge chosen in Line 2 is added to a set in $\mathcal{F}$, it is not in $\bigcup \mathcal{F}$ yet. So, the only way an edge can occur in two sets in $\mathcal{F}$ is if it is chosen in Line 2. Every set in $\mathcal{F}$ contains at most one such edge. Furthermore, if a set in $\mathcal{F}$ is defined starting from a blocked edge, it also contains another edge in A. Otherwise, the edge would not be blocked. So, we can choose, from every set in $\mathcal{F}$, an edge that occurs in no other set in $\mathcal{F}$.

By Lemma 7.7, there at most $\frac{n}{2}$ edges that are chosen in Line 2 at some point. The algorithm ensures that all $> nk$ edges in A are in $\bigcup \mathcal{F}$. So, each of the remaining $> nk - \frac{n}{2}$ edges is contained in a set in $\mathcal{F}$, and at most two of them are in the same set. So, there is a set D with $|D| > \frac{nk-\frac{n}{2}}{2}$ as required in the definition. □

The algorithm is correct and produces a set of cycles that induce automorphisms of $\mathfrak{G}^T$. This set yields the bound on the size of the orbit of α.

Proof of Lemma 7.5. By Lemma 7.8, there is a set of automorphisms such that each of them flips some edge occurring in α that is not flipped by any of the other automorphisms in the set. Thus, for every subset of these automorphisms, their concatenation yields a unique set of flipped edges in α, i.e., no two of these concatenations map α to the same tuple. So, $|\text{Orbit}(\alpha)| > 2^{\frac{nk-\frac{n}{2}}{2}}$, and, since $\frac{nk-\frac{n}{2}}{2} \geq \frac{nk-\frac{nk}{2}}{2} = \frac{nk}{4}$, it follows that $|\text{Orbit}(\alpha)| > 2^{\frac{nk}{4}}$. □

This completes the proof of Lemma 7.3.

We have established that it suffices to consider strong supports that are molecules. The winning strategy for Duplicator in the game on the strongly supported objects will be devised from a winning strategy in the game on the CFI-graphs themselves where Spoiler puts pebbles on strong supports. We first describe a winning strategy in the game on the CFI-graphs, where Spoiler may put pebbles on a constant number of molecules.

In any position of the game on CFI-graphs over $\mathcal{K}_n^<$, we call a vertex v (edge e) of $\mathcal{K}_n^<$ pebbled if there is a pebble on a vertex in the corresponding gadget v^* (e^*). A vertex or edge that is not pebbled is *free*. Further, a vertex v is $\geq \frac{n}{2}$-free if it is free and incident to $\geq \frac{n}{2}$ free edges that lead to free vertices. Otherwise, v is $< \frac{n}{2}$-free.

In case the game is played on non-isomorphic CFI-graphs, every bijection played by Duplicator has to map some edge in one CFI-graph to a non-edge in the other one. This error should always occur in a gadget adjacent to enough free edge gadgets. Note that, when Spoiler puts pebbles on the elements of a single molecule, this can already mean that each vertex is adjacent to a pebbled edge. We can, however, guarantee that the error always occurs in a $\geq \frac{n}{2}$-free vertex and can be "moved" to another $\geq \frac{n}{2}$-free vertex via free edges.

LEMMA 7.9. *Fix $\ell \in \mathbb{N}$, let $n \geq 12\ell$ and let $\mathfrak{G}^T$ be a CFI-graph over $\mathcal{K}_n^<$. If at most $n\ell$ edges and at most ℓ vertices of $\mathcal{K}_n^<$ are pebbled (with respect to a move of Spoiler in $\mathfrak{G}^T$), then there are $> \frac{n}{2}$ many vertices that are $\geq \frac{n}{2}$-free.*

PROOF. Assume otherwise. Then, at least $\frac{n}{2}$ vertices are not $\geq \frac{n}{2}$-free, i.e., they are either pebbled or incident to $\geq \frac{n}{2}$ pebbled edges. The least number of edge pebbles necessary for this situation is when all pebbled edges are again incident to the $< \frac{n}{2}$-free vertices that are not themselves pebbled. Since there are at most ℓ pebbled vertices, this means that the number of pebbled edges is

$$\begin{aligned}
&\geq \left(\frac{n}{2} - \ell\right)^2 \cdot \frac{1}{2} \\
&= \frac{n^2}{8} - \frac{4n\ell}{8} + \frac{\ell^2}{2} \\
&> \frac{n^2}{8} - \frac{4n\ell}{8} \\
&\geq \frac{12n\ell}{8} - \frac{4n\ell}{8} \\
&= n\ell.
\end{aligned}$$

This contradicts the assumption that at most $n\ell$ edges are pebbled. □

LEMMA 7.10. *Let $\ell \in \mathbb{N}$ and $n \geq 12\ell$. Further let $\mathcal{K}_n^< = (V, E, <)$ be the ordered complete graph over n vertices, and let $T = \emptyset$, $S = \{v\}$ for some vertex $v \in V$. Then, Duplicator wins the bijection game on the corresponding CFI-graphs $\mathfrak{G}^T$, $\mathfrak{G}^S$ where Spoiler has $n\ell$ pebbles that he may only place on edge gadgets, and ℓ additional pebbles that he may place only on vertex gadgets.*

Duplicator can further guarantee that for every bijection g she chooses, there is a set $F \subseteq E$, a vertex $v_0 \in V$ and an edge $e = \{v_0, v_1\}$ such that

- $g(x) = \begin{cases} \rho_{F\Delta\{e\}}(x), & \text{if } x \in v_0^*, \\ \rho_F(x), & \text{otherwise.} \end{cases}$
- *v_0 and v_1 are $\geq \frac{n}{2}$-free,*
- *e is free.*

This means that g preserves the parity of each vertex gadget except for v_0^*. We say that the pair (v_0, e) is *inconsistent*. If the parity of a vertex gadget is preserved, then the vertex is *consistent*.

PROOF. First, note that if Duplicator can always play a bijection with the above properties, she wins the bijection game: g preserves all edges except those between vertices in v_0^* and e^*, on which there are no pebbles.

Initially, Duplicator can play a bijection with the required properties by choosing $v_0 = v$ and $e \in E(v_0)$ arbitrarily. Now consider an arbitrary move for Duplicator in the bijection game such that she has previously played a bijection g with the required properties. Choose $F \subseteq E$, $v_0, v_1 \in V$ and $e \in E$ as above.

If, in the current move, v_0 and v_1 are still $\geq \frac{n}{2}$-free and e is still free, then Duplicator can simply play g again. Otherwise, we distinguish several cases.

(1) There is a pebble on v_0.
   (a) If v_1 is not $\geq \frac{n}{2}$-free anymore, then it is still incident to $\frac{n}{2} - 1$ free edges leading to free vertices, $> \frac{n}{2}$ of which are $\geq \frac{n}{2}$-free. So, v_1 has a $\geq \frac{n}{2}$-free neighbour v_2 such that $\{v_1, v_2\}$ is free. Since v_2 is $\geq \frac{n}{2}$-free, it is incident to at most $\frac{n}{2} - 1$ pebbled edges. Not counting v_2, there are still $\geq \frac{n}{2}$ many vertices that are $\geq \frac{n}{2}$-free, so one of them is connected to v_2 with a free edge.

      Call that vertex v_3, and choose $(v_2, \{v_2, v_3\})$ as the inconsistent pair for the new bijection g'. Let g' be the bijection associated with v_2, $\{v_2, v_3\}$ and the edge set $F' = F \triangle \{e, \{v_1, v_2\}, \{v_2, v_3\}\}$. Then, since $e \in F \triangle F'$, v_0 is now consistent. v_1 stays consistent, because an even number of edges incident to v_1 is in $F \triangle F'$.
   (b) If v_1 is still $\geq \frac{n}{2}$-free, then we choose a $\geq \frac{n}{2}$-free neighbour v_2 of v_1 and make the pair $(v_1, \{v_1, v_2\})$ inconsistent. Then, the edge set associated with the new bijection is $F \triangle \{e\}$.

(2) There is no pebble on v_0. Then, either there is a pebble on e, or v_0 or v_1 is not $\geq \frac{n}{2}$-free anymore. In all these cases, the new bijection can still be associated with the same edge set F. For the inconsistent pair, simply choose a new $\geq \frac{n}{2}$-free vertex v_0'. v_0' has a $\geq \frac{n}{2}$-free neighbour v_1' such that the edge $e' = \{v_0', v_1'\}$ is free. So, the inconsistent pair (v_0', e') makes the new bijection satisfy all desired properties. □

Transferring this winning strategy to the structure over strongly supported sets relies on a special representation of every strongly supported set as a combination of a strong support that is a molecule and a kind of template for the structure of the set. This representation is well-defined in case the CFI-graphs are homogeneous.

The definition of homogeneity, i.e., that tuples with the same type are mapped to each other by an automorphism, constitutes a small technical difficulty: The usual notion of $\mathbf{C}^m$-types is not easily applicable to the tuples that may form strong supports, because their length depends on the size of the input structure. To establish homogeneity in a useful sense, we therefore change the notion of types.

Definition 7.11. Fix $k \in \mathbb{N}$. The *type* $\mathrm{tp}(\alpha)$ of the tuple $\alpha = (a_1, \ldots, a_s)$ of length $s \in \mathbb{N}$ from the structure $\mathfrak{A}$ is the set of all $\mathbf{C}^{s+4k(4k-1)+3}$-formulas φ such that

$-\varphi$ has free variables $x_1, \ldots, x_s$,
$-x_1, \ldots, x_s$ **do not occur as bound variables in** ψ,
$-\mathfrak{A} \models \varphi(a_1, \ldots, a_s)$.

As the length of the tuple α is not fixed for fixed k, it can be any tuple, in particular a molecule or a concatenation of molecules. The number of variables is chosen in a way that CFI-graphs over ordered complete graphs are homogeneous with respect to this definition of types.

Lemma 7.12 (Homogeneity). *Let $k \in \mathbb{N}$, and let $\mathfrak{G}^T$ be a CFI-graph over $\mathcal{K}_n^< = (V, E, <)$. For any two molecules α, γ of the same length,* $\mathrm{tp}(\alpha) = \mathrm{tp}(\gamma)$ *implies that there is an automorphism of $\mathfrak{G}^T$ mapping α to γ.*

Proof. Induction on the length i of the molecules α and γ. The statement is clear for $i = 0$, and for $i = 1$, it follows from the fact that the gadgets are definable via the linear order on the

underlying graph. In the induction step, let $i \geq 1$ and consider $(i+1)$-tuples αa, γc with the same type.

Since also $\mathrm{tp}(\gamma) = \mathrm{tp}(\alpha)$, there is an automorphism π with $\pi(\gamma) = \alpha$ by induction hypothesis. Choose $b = \pi(c)$. it suffices to show that there is an automorphism ρ with $\rho(\alpha a) = \alpha b$, because then, $\pi^{-1}\rho$ is an automorphism mapping αa to γc.

Since αa and γc are molecules with at least two elements, they consist only of vertices from edge gadgets. Each edge gadget is definable in C^3 using the order on the vertex gadgets, so, as $\mathrm{tp}(\alpha a) = \mathrm{tp}(\gamma c) = \mathrm{tp}(\alpha b)$, a and b are in the same edge gadget e^*.

We call an edge gadget (or its corresponding edge) *free* if no vertex from the gadget occurs in α (respectively, γ), and otherwise the gadget and corresponding edge are *fixed*. A vertex is $\geq \frac{n}{2}$*-free* if it is incident to $\geq \frac{n}{2}$ free edges, and $< \frac{n}{2}$*-free* otherwise.

(1) First, consider the case that there are disjoint paths via free edges from both vertices incident to e to $\geq \frac{n}{2}$-free vertices u and v. If $u = v$ or the edge $\{u, v\}$ is free, then it follows immediately that e is on a cycle consisting of free edges. If the edge $\{u, v\}$ is fixed, then u and v are each incident to at least $\frac{n}{2}$ free edges to the remaining $n-2$ vertices. So, there is a vertex w that is connected to both u and v with a free edge, and e is again on a cycle of free edges. Let F be the set of edges on that cycle. Then, ρ_F is an automorphism of $\mathfrak{G}^T$ that fixes α and exchanges a and b.

(2) If there are no such disjoint paths, then assume there is no automorphism exchanging a and b (i.e., flipping e). We construct a formula with $4k(4k-1)+3$ additional variables that distinguishes a and b.

Choose u and v such that $e = \{u, v\}$ and all vertices reachable from u without using e or a fixed edge are $< \frac{n}{2}$-free. If no such u exists, then we are in Case 1.

Since there is no automorphism flipping e, there is a unique element of $e^* = \{a, b\}$ such that the gadget u^* is even if, and only if, that element is labelled e^1. Our formula will state that property for all automorphisms flipping only edges incident to $< \frac{n}{2}$-free vertices. Since there is a constant number of $< \frac{n}{2}$-free vertices, we can deduce a bound on the number of variables.

Let H be the subgraph of G induced by the free vertices reachable from u without using e or a fixed edge. Then, the image of any edge gadget incident to u under any automorphism is uniquely determined by an automorphism that flips only edges in H. So, we can quantify over all relevant automorphisms by quantifying over the free edges in H. Furthermore, all vertices in H are $< \frac{n}{2}$-free.

For ease of notation, we fix an isomorphic copy of $\mathfrak{G}^T$ such that all gadgets corresponding to vertices in H are even. Let $e_1 \dots e_h$ be an enumeration of the edges in H. For any $e \in E$, let $\varphi_{e^*}(x)$ be the formula expressing that $x \in e^*$, and for $v \in V$, let $\varphi_{v^*}(y)$ state that $y \in v^*$. Then, the following formula defines the desired property:

$$\psi \coloneqq \forall x_1 \dots x_h \left(\bigwedge_{1 \leq i \leq h} \varphi_{e_i^*}(x_i) \wedge \forall y \left(\bigvee_{v \in V[H] \setminus \{u\}} \varphi_{v^*}(y) \rightarrow \varphi_{\mathrm{even}}(x_1, \dots, x_h, y) \right) \right.$$
$$\left. \rightarrow \forall y (\varphi_{u^*}(y) \rightarrow \varphi_{\mathrm{even}}(x_1, \dots, x_h, y)) \right).$$

φ_{even} should hold for exactly those vertices that are connected to an even number of edge vertices that are either in the tuple $x_1, \dots, x_h$, occur in α, or are replaced for the free variable x.

Let $F^1 = \{f \in E : f^1 \text{ occurs in } \alpha\}$ and $F^0 = \{f \in E : f^0 \text{ occurs in } \alpha\}$. For any $f \in F^1 \cup F^0$, let a_f be the corresponding entry of α. Then, φ_{even} is the formula

$$\bigvee_{2 \le i \le \lfloor \frac{n-1}{2} \rfloor} \exists^{2i} z \left(Eyz \wedge \left(z = x \vee \bigvee_{1 \le i \le h} z = x_i \vee \bigvee_{f \in F^1} z = a_f \vee \bigvee_{f \in F^0} \left(\varphi_{f^*}(z) \wedge z \neq a_f \right) \right) \right).$$

Every automorphism flipping an edge incident to u corresponds to an assignment of the variables $x_1, \ldots, x_h$ to vertices in the edge gadgets in H that make all vertex gadgets in H even. So, the formula indeed defines the unique vertex from e^* that, if set to e^1, makes u^* an even gadget.

The variables used in ψ are $x_1, \ldots, x_h, x, y, z$. Note that, for the formulas of the form φ_{e^*} or φ_{v^*}, it suffices to reuse z.

The number h of edges in H can be computed from the number of $< \frac{n}{2}$-free vertices in the whole graph.

We claim that at most $4k$ vertices are $< \frac{n}{2}$-free. Every $< \frac{n}{2}$-free vertex is incident to at least $\frac{n}{2}$ fixed edges. So, if there were $> 4k$ such vertices, then there would be

$$> \frac{1}{2} \cdot \frac{n}{2} \cdot 4k = nk$$

fixed vertices, which is not possible, because α is a molecule.

It follows that $h \le 4k(4k-1)$. So, the number of variables in ψ is $\le 4k(4k-1) + 3$. □

Next, we give a definition of the representation of sets in terms of their supports introduced by Dawar et al. (2008). This definition can directly be transferred to sets supported by molecules and the modified type definition.

Definition 7.13. Let c be a constant symbol. The set of *forms* is the least set containing c and every finite set of pairs (φ, τ) where φ is a form and $\tau = \text{tp}(\alpha\beta)$ for molecules α, β.

Definition 7.14. The *denotation* of a form φ and a molecule α over a structure $\mathfrak{A}$ is defined as follows:

— $c \star \alpha = \alpha_1$ (in particular, $c = a$ if $\alpha = (a)$),
— $\varphi \star \alpha = \{\psi \star \beta : (\psi, \text{tp}(\alpha, \beta)) \in \varphi\}$ if φ is a set.

Lemma 7.15 ((Dawar et al. 2008)). *α supports $\varphi \star \alpha$.*

The following useful property of denotations follows from the homogeneity condition:

Lemma 7.16 ((Dawar et al. 2008)). *$\pi(\varphi \star \alpha) = \varphi \star \pi(\alpha)$ for any automorphism π.*

The denotations of forms and k-tuples (instead of molecules), as defined by Dawar et al. (2008), are exactly the objects with a support of size k. The precise characterisation does not transfer to strong supports. Nevertheless, every strongly supported object is the denotation of a form and a strong support.

Lemma 7.17. *Let $x \in \text{HF}(\mathfrak{A})_{k-\text{ss}}$ for a CFI-graph $\mathfrak{A}$ over $\mathcal{K}_n^<$. Then, there is a form φ and a molecule α such that $x = \varphi \star \alpha$ and α strongly supports x.*

Proof. If x is an atom, then it is strongly supported by the molecule $\alpha := (x)$, and $x = c \star \alpha$. If x is a set, then choose a molecule α that strongly supports x. Analogously to the proof of Lemma 38

in Dawar et al. (2008), one can show that there is a form φ such that $x = \varphi \star \alpha$, using our modified homogeneity condition from Lemma 7.12. □

When transferring winning strategies, we will lift bijections on molecules to bijections on strongly supported sets. The lifted bijections are easier to define if the image of every strongly supported set depends only on the image of a single, canonical support.

Lemma 7.18. *For every linear order on the set of forms and every linear order on the set of molecules, every $x \in \mathrm{HF}(\mathfrak{A})$ has a unique representation (φ_x, α_x) such that*

—$\varphi_x \star \alpha_x = x$,
—α_x strongly supports x,
—$\mathrm{tp}(\alpha_x)$ is the minimal (w.r.t. the given linear order) type of a molecule strongly supporting x,
—φ_x is the minimal (w.r.t. the given linear order) form such that $\varphi_x \star \alpha = x$ for any α of type τ.

We call the pair (φ_x, α_x) the canonical representation *of x.*

Proof. Given linear orders on types and molecules, it is easy to define a minimal form φ_x and a molecule α_x of minimal type satisfying the conditions from the lemma. It remains to show that α_x is unique.

Suppose there are molecules $\alpha \neq \beta$ of type τ such that $\varphi_x \star \alpha = \varphi_x \star \beta = x$ and both α and β are strong supports. By the homogeneity condition, there is an automorphism π mapping α to β, because they are of the same type. But then $\pi(x) = \pi(\varphi_x \star \alpha) = \varphi_x \star \pi(\alpha) = \varphi_x \star \beta = x$, even though $\pi(\alpha) \neq \alpha$. This contradicts the fact that α is a strong support. So, α_x is unique. □

The representation in terms of forms and molecules further yields a characterisation of the set structure of $\mathrm{HF}(\mathfrak{A})_{k-\mathrm{ss}}$ based solely on the form and the type of the molecule. This will ensure that the lifted bijections preserve the set structure, since the underlying bijections obtained from the game on the CFI-graphs will preserve the types of molecules.

Lemma 7.19 ((Dawar et al. 2008)). *There are relations* In *and* Eq *such that, for all forms φ, ψ and all molecules α, β of the same length, $\varphi \star \alpha = \psi \star \beta$ if, and only if, $\mathrm{Eq}(\varphi, \psi, \mathrm{tp}(\alpha, \beta))$ and $\varphi \star \alpha \in \psi \star \beta$ if, and only if, $\mathrm{In}(\varphi, \psi, \mathrm{tp}(\alpha, \beta))$.*

The proof is completely analogous to that of Lemma 39 in Dawar et al. (2008), the modified definition of types does not change the definition of the relations In and Eq.

We now use that machinery together with the winning strategy in the game on the CFI-graphs to describe a winning strategy in the game on the strongly supported sets.

Lemma 7.20. *Let $k, m \in \mathbb{N}$, $\ell = km + 4k(4k-1) + 3$, $n \geq 12\ell$, and let $\mathfrak{G}^{\emptyset}, \mathfrak{G}^{\{v\}}$ be CFI-graphs over $\mathcal{K}_n^<$ for some vertex v. Then, Duplicator wins the m-pebble bijection game on $\mathrm{HF}(\mathfrak{G}^{\emptyset})_{k-\mathrm{ss}}$ and $\mathrm{HF}(\mathfrak{G}^{\{v\}})_{k-\mathrm{ss}}$.*

Consider any position $(x_1, y_1), \ldots, (x_m, y_m)$ in the game. We will later use as an invariant that there are forms $\varphi_1, \ldots, \varphi_m$ and molecules $\alpha_1, \ldots, \alpha_m$, $\beta_1, \ldots, \beta_m$, such that $x_i = \varphi_i \star \alpha_i$, $y_i = \varphi_i \star \beta_i$ and Duplicator wins the bijection game from the position $(\alpha_1, \beta_1), \ldots, (\alpha_m, \beta_m)$ if Spoiler can place $n\ell$ pebbles on edge gadgets and ℓ pebbles on vertex gadgets (where the pair (α_i, β_i) means that there are pebbles on all corresponding pairs of entrys of the molecules α_i, β_i). We use Duplicator's winning strategy in the game on the structures $\mathfrak{G}^{\emptyset}$ and $\mathfrak{G}^{\{v\}}$ to obtain a bijection g for Duplicator in the game on the strongly supported sets.

Consider a position in the game on $\mathfrak{G}^{\emptyset}, \mathfrak{G}^{\{v\}}$ with pebbles on $(\alpha_2, \beta_2), \ldots, (\alpha_m, \beta_m)$. Assume Duplicator has a winning strategy with $n\ell > nkm$ pebbles on edge gadgets and $\ell > km$ pebbles on vertex gadgets (which will follow from Lemma 7.10 by induction). Then, there is a bijection

f between molecules such that Duplicator wins from the position $(\alpha, f(\alpha)), (\alpha_2, \beta_2), \ldots, (\alpha_m, \beta_m)$ for every molecule α: If Spoiler successively pebbles the elements $a_1, \ldots, a_{nk}$ of α, then Duplicator plays bijections $h_1, h_2[a_1], h_3[a_1, a_2], \ldots, h_{nk}[a_1, \ldots, a_{nk-1}]$ according to her winning strategy. (If the molecules are tuples of length $s < nk$, then fix any additional vertices $a_{s+1}, \ldots, a_{nk}$ from edge gadgets.)

Choose $f(\alpha) = (h_1(a_1), h_2[a_1](a_2), h_3[a_1, a_2](a_3), \ldots, h_{nk}[a_1, \ldots, a_{nk-1}](a_{nk}))$. Note that f is a bijection, because its inverse can be constructed using the same bijections $h_1, h_2[a_1], \ldots, h_{nk}[a_1, \ldots, a_{nk-1}]$. Define $g : \mathrm{HF}(\mathfrak{G}^{\emptyset})_{k-\mathrm{ss}} \to \mathrm{HF}(\mathfrak{G}^{\{v\}})_{k-\mathrm{ss}}$ through $g(x) = \varphi_x \star f(\alpha_x)$, where (φ_x, α_x) is the canonical representation of x. We show that g is well-defined and a bijection.

Lemma 7.21. *For every automorphism π and every molecule α, $\mathrm{tp}(\alpha, \pi(\alpha)) = \mathrm{tp}(f(\alpha), \pi(f(\alpha)))$, where we identify each automorphism $\pi = \rho_F$ of $\mathfrak{G}^{\emptyset}$ with the automorphism of $\mathfrak{G}^{\{v\}}$ induced by the same edge set F.*

Proof. By definition of g, Duplicator wins the $4k(4k-1)+3$-pebble bijection game on $\mathfrak{G}^{\emptyset}, \mathfrak{G}^{\{v\}}$ while always mapping α to $f(\alpha)$. But this also means that, for every entry α_i of α, $f(\alpha)_i$ already uniquely determines how her bijection acts on the whole gadget containing α_i.

> α consists only of edge gadgets. In this case, $\mathrm{tp}(\alpha, \pi(\alpha))$ is already completely determined by equalities. If $\pi(\alpha_i) = \alpha_i$, then clearly also $\pi(f(\alpha)_i) = f(\alpha)_i$. Otherwise, $\alpha_i = e^j$ and $\pi(\alpha_i) = e^{1-j}$ for some $j \in \{0, 1\}$. But, since $f(\alpha)_i$ can be either e^j or e^{1-j}, π also flips $f(\alpha)_i$.
>
> $\alpha = v^X$ for some vertex v. Now $\mathrm{tp}(\alpha, \pi(\alpha))$ is completely determined by the symmetric difference of the vertices from edge gadgets adjacent to v^X and $\pi(v^X)$. Let $\pi(\alpha) = v^Y$ and $f(\alpha) = v^{X \triangle Z}$. Assume $v^{Y'} := f(\pi(\alpha)) \neq v^{Y \triangle Z}$. Then, there is some edge $e \in (Y \triangle Y') \setminus Z$. Then, Spoiler wins by putting a pebble on v^Y and then on e^1: Since $e \in (Y \triangle Y') \setminus Z$, e^1 is incident to v^X if, and only if, it is incident to $v^{X \triangle Z} = f(v^X)$. But e^1 is incident to v^Y if, and only if, it is *not* incident to $v^{Y'}$. □

Lemma 7.22. *g is well-defined.*

Proof. We need to show that for every set $\varphi \star \alpha$ that is strongly supported by α, $\varphi \star f(\alpha)$ is also strongly supported by $f(\alpha)$.

Let π be an automorphism of $\mathfrak{G}^{\emptyset}$ and $\mathfrak{G}^{\{v\}}$, where we identify automorphisms as in Lemma 7.21.

By Lemma 7.21, $\mathrm{tp}(\alpha, \pi(\alpha)) = \mathrm{tp}(f(\alpha), \pi(f(\alpha)))$. Since $\pi(\varphi \star f(\alpha)) = \varphi \star \pi(f(\alpha))$, it follows that

$$\begin{aligned}
&\varphi \star \alpha = \pi(\varphi \star \alpha) \\
\Leftrightarrow\ &\varphi \star \alpha = \varphi \star \pi(\alpha) \\
\Leftrightarrow\ &\mathrm{Eq}(\varphi, \varphi, \mathrm{tp}(\alpha, \pi(\alpha))) \\
\Leftrightarrow\ &\mathrm{Eq}(\varphi, \varphi, \mathrm{tp}(f(\alpha), \pi(f(\alpha)))) \\
\Leftrightarrow\ &\varphi \star f(\alpha) = \varphi \star \pi(f(\alpha)) \\
\Leftrightarrow\ &\varphi \star f(\alpha) = \pi(\varphi \star f(\alpha)).
\end{aligned}$$

So, π fixes $\varphi \star \alpha$ if, and only if, π fixes $\varphi \star f(\alpha)$. Since, by definition of molecules, π also fixes α if, and only if, it fixes $f(\alpha)$, it follows that α strongly supports $\varphi \star \alpha$ if, and only if, $f(\alpha)$ strongly supports $\varphi \star f(\alpha)$. □

Lemma 7.23. *For every set x that is strongly supported by a molecule α, (φ, α) is the canonical representation of $x = \varphi \star \alpha$ if, and only if, $(\varphi, f(\alpha))$ is the canonical representation of $\varphi \star f(\alpha)$.*

Proof. Since f is a bijection, the proofs for both directions are symmetric. Let $\varphi \star \alpha$ be canonical, and suppose that $\varphi \star f(\alpha)$ is not canonical. Then, $\mathrm{tp}(f(\alpha))$ is not minimal, or it is minimal, but φ is not.

$\mathrm{tp}(f(\alpha))$ is not minimal:. Then, there is a molecule β with $\mathrm{tp}(f(\beta)) < \mathrm{tp}(f(\alpha))$ such that $f(\beta)$ strongly supports $\varphi \star f(\alpha)$. Since f preserves types of molecules, also $\mathrm{tp}(\beta) < \mathrm{tp}(\alpha)$. Furthermore, β strongly supports $\varphi \star \alpha$: Let π be an automorphism. Then,

$$\begin{aligned} & \pi(\varphi \star \alpha) = \varphi \star \alpha \\ \Leftrightarrow\ & \pi(\alpha) = \alpha \\ \Leftrightarrow\ & \pi(f(\alpha)) = f(\alpha) \\ \Leftrightarrow\ & \pi(\varphi \star f(\alpha)) = \varphi \star f(\alpha) \\ \Leftrightarrow\ & \pi(f(\beta)) = f(\beta) \\ \Leftrightarrow\ & \pi(\beta) = \beta, \end{aligned}$$

since $f(\beta)$ strongly supports $f(\alpha)$, and α and $f(\alpha)$ (respectively, β and $f(\beta)$) are fixed by the same automorphisms (i.e., those we identify because they flip the same edges).

φ is not minimal:. Then, there are a form ψ and a tuple β such that $\psi < \varphi$ and $\psi \star f(\beta) = \varphi \star f(\alpha)$. Since $\mathrm{tp}(f(\alpha))$ is minimal, the tuples $f(\beta), f(\alpha), \alpha$ and β all have the same type.

By the homogeneity condition shown in Lemma 7.12, there is an automorphism π that maps $f(\alpha)$ to $f(\beta)$ and α to β. So, by Lemma 7.21, $\mathrm{tp}(f(\alpha), f(\beta)) = \mathrm{tp}(\alpha, \beta)$. But then, because of the relation Eq, $\varphi \star f(\alpha) = \psi \star f(\beta)$ implies $\varphi \star \alpha = \psi \star \beta$, which contradicts the assumption that (φ, α) is canonical. □

Lemma 7.24. *g is a bijection.*

Proof. First, we show that g is injective. Let $\varphi \star f(\alpha) = \psi \star f(\beta)$, such that $f(\alpha), f(\beta)$ are strong supports and (φ, α) and (ψ, β) are canonical representations. By Lemma 7.23, $(\varphi, f(\alpha))$ and $(\psi, f(\beta))$ are also canonical representations. So, $\varphi = \psi$ and $f(\alpha) = f(\beta)$, and thus $\varphi \star \alpha = \psi \star \beta$.

To show surjectivity, let $x \in \mathrm{HF}(\mathfrak{B})$ be strongly supported, and let (φ, α) be its canonical representation. By Lemma 7.23, $(\varphi, f^{-1}(\alpha))$ is also canonical. So, $\varphi \star \alpha$ is in the image of g (note that, analogously to the well-definedness proof, $\varphi \star f^{-1}(\alpha)$ is strongly supported by $f^{-1}(\alpha)$). □

Proof of Lemma 7.20. We show that Duplicator wins the game by playing the bijections g defined above. With that strategy, she maintains the following invariant: In every position $(x_1, y_1), \ldots, (x_m, y_m)$, there are forms $\varphi_1, \ldots, \varphi_m$ and molecules $\alpha_1, \ldots, \alpha_m, \beta_1, \ldots, \beta_m$ such that, for every i, (φ_i, α_i) is the canonical representation of x_i and (φ_i, β_i) is the canonical representation of y_i, and Duplicator wins the bijection game from the position $(\alpha_1, \beta_1), \ldots, (\alpha_m, \beta_m)$ if Spoiler can place $n\ell$ pebbles on edge gadgets and ℓ pebbles on vertex gadgets.

In the initial position, we assume that all pebbles are on $\emptyset$, and choose all forms and molecules to be empty. Duplicator wins the game by Lemma 7.10. For the induction step, let $(x_1, y_1), \ldots, (x_m, y_m)$ be the position after Duplicator has played the bijection g and Spoiler has placed a pebble on the pair $(x_1, g(x_1) = y_1)$. Let (φ_1, α_1) be the canonical representation of x_1 and choose $\beta_1 = f(\alpha_1)$. Since f is played as part of a winning strategy, and by Lemmas 7.22 and 7.23, the invariant holds.

It remains to show that $x_1, \ldots, x_m \mapsto y_1, \ldots, y_m$ is a partial isomorphism.

The edge relation between atoms is preserved, because Duplicator wins the game on the strong supports.

Spoiler only needs nkm pebbles on edge gadgets and km pebbles on vertex gadgets to fix the strong supports of $x_1, \ldots, x_m$ and $y_1, \ldots, y_m$. Since $\ell = km + 4k(4k-1) + 3$, there are $\geq 4k$

$(4k-1)+3$ pebbles for both edge and vertex gadgets left to verify that $\mathrm{tp}(\alpha_1,\ldots,\alpha_m) = \mathrm{tp}(\beta_1,\ldots,\beta_m)$. It follows that the relations $=$ and $\in$ between sets are preserved because of the relations In and Eq:

$$\begin{aligned} & x_i = x_j \\ \Leftrightarrow\ & \mathrm{Eq}(\varphi_i,\varphi_j,\mathrm{tp}(\alpha_i,\alpha_j)) \\ \Leftrightarrow\ & \mathrm{Eq}(\varphi_i,\varphi_j,\mathrm{tp}(\beta_i,\beta_j)) \\ \Leftrightarrow\ & \varphi_i \star \beta_i = \varphi_j \star \beta_j \\ \Leftrightarrow\ & y_i = y_j. \end{aligned}$$

The proof for $\in$ is analogous. □

Theorem 7.25. *Let $\Pi \in \mathrm{CPT}$ be a program that activates only strongly supported sets. Then, Π cannot decide the Cai-Fürer-Immerman query over ordered complete graphs.*

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

Let $n \mapsto n^q$ for some $q \geq 1$ be the resource bound of Π, and let $k = 4q$. By Lemma 7.3, every object activated by Π in the run on a CFI-graph $\mathfrak{G}^T$ over $\mathcal{K}_n^<$ has a strong support that consists of $< nk$ vertices that are all in edge gadgets. Hence, $\mathrm{Active}(\Pi,\mathfrak{G}^T) \subseteq \mathrm{HF}(\mathfrak{G}^T)_{k-\mathrm{ss}}$ for every such CFI-graph.

Let $\ell = km + 4k(4k-1)+3$ and let $\mathfrak{G}^T$ be an even and $\mathfrak{G}^S$ be an odd CFI-graph over the ordered complete graph $\mathcal{K}_n^<$ for some $n \geq 12\ell$. By Lemma 7.20, Duplicator wins the m-pebble bijection game on $\mathrm{HF}(\mathfrak{G}^T)_{k-\mathrm{ss}}$ and $\mathrm{HF}(\mathfrak{G}^S)_{k-\mathrm{ss}}$. So, Π cannot distinguish between $\mathfrak{G}^T$ and $\mathfrak{G}^S$. □

This result provides a deeper understanding of the expressive power of fragments of Choiceless Polynomial Time. Interpretation Logic, which was introduced by Grädel et al. (2015), 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 Grädel et al. (2015)), 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 $\mathrm{while}_{\mathrm{new}}$, which permits construction of new elements for definable *tuples*. For the fragment with counting, it could be shown (Grädel et al. 2015) that, on structures with bounded colour class size, only CPT computations with sets of bounded rank can be simulated. Theorem 7.25 now implies that the fragment not only fails to define sets of unbounded rank, but also any kind of setlike objects.

Corollary 7.26. *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 transitively 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 three-regular graphs, appear 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 setlike objects separates two of the few known natural fragments of CPT. Our characterisation of setlike objects could provide new insights for other sequence-based formalisms.

REFERENCES

F. Abu Zaid, E. Grädel, M. Grohe, and W. Pakusa. 2014. Choiceless polynomial time on structures with small abelian colour classes. In *Proceedings of the International Symposium on Mathematical Foundations of Computer Science (MFCS'14)*. LNCS, Vol. 8634. Springer, 50–62.

A. Blass. 2008. Why sets? In *Pillars of Computer Science*. LNCS, Vol. 4800. Springer, 179–198. DOI: http://dx.doi.org/10.1007/978-3-540-78127-1_11

A. Blass, Y. Gurevich, and S. Shelah. 1999. Choiceless polynomial time. *Ann. Pure Appl. Logic* 100, 1 (1999), 141–187.

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

A. Blass, Y. Gurevich, and J. Van den Bussche. 2000. Abstract state machines and computationally complete query languages. In *Proceedings of the International Workshop on Abstract State Machines*. Springer, 22–33.

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

A. Chandra and D. Harel. 1982. Structure and complexity for relational queries. *J. Comput. Syst. Sci.* 25 (1982), 99–128.

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

A. Dawar, M. Grohe, B. Holm, and B. Laubner. 2009. Logics with rank operators. In *Proceedings of the Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'09)*. 113–122.

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

E. Grädel and M. Grohe. 2015. Is polynomial time choiceless? In *Fields of Logic and Computation II (LNCS)*, Vol. 9300. Springer, 193–209.

E. Grädel, Ł. Kaiser, W. Pakusa, and S. Schalthöfer. 2015. Characterising choiceless polynomial time with first-order interpretations. In *Proceedings of the Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'15)*.

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

M. Grohe. 2008. The quest for a logic capturing PTIME. In *Proceedings of the Conference on Logic in Computer Science, (LICS'08)*. IEEE, 267–271.

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

N. Immerman. 1986. Relational queries computable in polynomial time. *Inf. Control* 68 (1986), 86–104.

B. Laubner. 2011. *The Structure of Graphs and New Logics for the Characterization of Polynomial Time*. Ph.D. Dissertation. HU Berlin.

W. Pakusa, S. Schalthöfer, and E. Selman. 2016. Definability of Cai-Fürer-Immerman problems in choiceless polynomial time. In *Proceedings of the 25th EACSL Annual Conference on Computer Science Logic (CSL'16) (Leibniz International Proceedings in Informatics (LIPIcs))*, Vol. 62. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 19:1–19:17.

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

M. Y. Vardi. 1982. The complexity of relational query languages. In *Proceedings of the ACM Symposium on Theory of Computing (STOC'82)*. ACM Press, 137–146.

Received March 2017; accepted October 2017