Lower Bounds for Choiceless Polynomial Time via Symmetric XOR-Circuits

Benedikt Pago
Mathematical Foundations of Computer Science, RWTH Aachen University, Germany

Abstract

Choiceless Polynomial Time (CPT) is one of the few remaining candidate logics for capturing PTIME. In this paper, we make progress towards separating CPT from polynomial time by firstly establishing a connection between the expressive power of CPT and the existence of certain symmetric circuit families, and secondly, proving lower bounds against these circuits. We focus on the isomorphism problem of unordered Cai-Fürer-Immerman-graphs (the CFI-query) as a potential candidate for separating CPT from PTIME. Results by Dawar, Richerby and Rossman, and subsequently by Pakusa, Schalthöfer and Selman show that the CFI-query is CPT-definable on linearly ordered and preordered base graphs with small colour classes. We define a class of CPT-algorithms, that we call "CFI-symmetric algorithms", which generalises all the known ones, and show that such algorithms can only define the CFI-query on a given class of base graphs if there exists a family of symmetric XOR-circuits with certain properties. These properties include that the circuits have the same symmetries as the base graphs, are of polynomial size, and satisfy certain fan-in restrictions. Then we prove that such circuits with slightly strengthened requirements (i.e. stronger symmetry and fan-in and fan-out restrictions) do not exist for the n-dimensional hypercubes as base graphs. This almost separates the CFI-symmetric algorithms from PTIME – up to the gap that remains between the circuits whose existence we can currently disprove and the circuits whose existence is necessary for the definability of the CFI-query by a CFI-symmetric algorithm.

2012 ACM Subject Classification Theory of computation → Finite Model Theory

Keywords and phrases logic in computer science, finite model theory, descriptive complexity, symmetric computation, symmetric circuits, graph isomorphism

Digital Object Identifier 10.4230/LIPIcs.MFCS.2023.73

Related Version *Full Version*: `https://arxiv.org/abs/2302.05426` [27]

Acknowledgements I thank Daniel Wiebking for his group-theoretic help.

1 Introduction

A central open question in finite model theory is whether there exists a logic that captures the complexity class polynomial time. It was first raised by Chandra and Harel in 1980 [5] and later made precise by Gurevich [20]. According to his definition, the term "logic" refers to any computation model that operates on finite structures and is *isomorphism-invariant*, i.e. yields the same output on isomorphic input structures. The question for a logic for PTIME thus asks whether there exists some isomorphism-invariant computation model which can decide exactly the same classes of structures that can be decided by "classical" polynomial time algorithms (i.e. Turing machines). The latter are not isomorphism-invariant because every structure, for example a graph, has multiple different representations as a binary string (e.g. depending on the vertex order that is used for the adjacency list/matrix). The result of a classical computation depends on this string representation and not on the isomorphism-type of the graph, which is undesirable for a logic. Another way of phrasing the question is whether the time and space cost of ensuring symmetry-invariance in computations is necessarily super-polynomial or not. Gurevich himself conjectured that no logic for PTIME exists, and if we had a proof for this, it would immediately separate P and NP: It is long known by Fagin's

theorem [14] that NP is captured by existential second order logic. In fact, many candidate logics that have been proposed in an attempt to capture PTIME have been proven to be only a strict fragment of it. For a survey on this topic (excluding the results from recent years), see [18].

One prominent logic for which research on lower bounds has not been so successful yet is *Choiceless Polynomial Time* (CPT). It was introduced in 1999 by Blass, Gurevich and Shelah [3] as a symmetry-invariant machine model. It has been open since then whether CPT (with counting) does capture PTIME or not. For more background information on CPT, see for example [17, 28, 32]. CPT can also be viewed as an extension of *fixed-point logic with counting* [6] with *hereditarily finite sets* as data structures. Other (perhaps less studied) candidates that have not been separated from PTIME so far are logics with witnessed choice constructs (a concept first introduced in [15]). These include fixed-point logic with witnessed symmetric choice and interpretations [23] and CPT extended with witnessed symmetric choice [24]. Prior to Lichter's breakthrough [22], which separates *rank logic* from PTIME using a variation of the famous Cai-Fürer-Immerman (CFI) construction [4], logics with *linear-algebraic operators* [7] were also considered reasonable candidates. However, as outlined in [8], the results from [22] and [7] together imply that *no* set of isomorphism-invariant linear algebraic operators can be used to define a logic capturing PTIME. Thus, an important next step in this program would be to also rule out CPT as a logic for PTIME. In this article, we make progress towards this goal and propose an approach that allows to infer CPT lower bounds from lower bounds against certain symmetric Boolean circuit families. Thereby, the problem is narrowed down to the study of concrete combinatorial objects, for which we can present a first lower bound.

It should be remarked that the power of CPT is also of interest in another research context, namely with regards to the *graph isomorphism problem*: One can roughly divide the most important graph isomorphism algorithms into group-theoretic and combinatorial ones; the latter term refers to generalisations of the well-known *Weisfeiler-Leman* (WL) method, and these are choiceless. It is in a sense possible to characterise CPT as the class of all polynomial-time combinatorial graph isomorphism algorithms. A precise description of these has been given with the *Deep Weisfeiler-Leman* computation model [19], which was shown to be equivalent to CPT. Hence, lower bounds against CPT would also imply limitations for all combinatorial graph isomorphism algorithms.

Concerning CPT lower bounds, only relatively little is known. There is a non-definability result for a *functional* problem in PTIME, namely, it is impossible to define the dual of a given finite vector space in CPT [30]. What we would like to have is, however, the inexpressibility of a polynomial-time *decision problem*. We focus on a standard benchmark from finite model theory, namely the *CFI-query* [4]. Instances of this query are obtained by applying the so-called CFI construction to any connected undirected *base graph*, yielding either an "even" or an "odd" CFI-graph (see Section 3). The query asks to determine the parity of a given CFI-graph and it can be seen as a variant of the graph isomorphism problem or as the problem of solving a certain linear equation system over the finite field $\mathbb{F}_2$. Its descriptive complexity depends highly on the choice of base graphs and whether these come with a built-in linear order or not. The CFI-query is decidable in polynomial time but not in fixed-point logic with counting [4]. It is open whether it is CPT-definable on *unordered* base graphs, and our goal is to eventually answer this question in the negative. Our approach starts off from positive results: There do exist CPT-algorithms for linearly ordered and preordered versions of the CFI-query [9, 29] and also CFI-graphs over base graphs of linear degree [29]. In [26] it was shown that there exist unordered CFI-graphs (over n-dimensional hypercubes) whose degree is not linear and which cannot be preordered in CPT in such a way that the preorder-based algorithm from [29] (or the total-order-based one from [9]) could

be applied. This shows that these known choiceless algorithms for preordered versions of the CFI-query do not generalise to the unordered case because the necessary combinatorial objects (said preorders) are not symmetric enough. In the present paper, we define a general class of CPT-algorithms for the CFI-query, which encompasses all the known ones mentioned above, and show that their expressiveness depends on the existence of certain symmetric combinatorial objects, namely circuits with Boolean XOR-gates. We show that the CFI-query over a given class $\mathcal{K}$ of base graphs is only definable by an algorithm from that class if there exists a family of polynomial-size symmetric XOR-circuits with the properties in Theorem 1. This means that the non-definability of the CFI-query over $\mathcal{K}$ can be shown by proving the non-existence of such circuits. In Theorem 2, we almost achieve this goal: If we take as $\mathcal{K}$ the family of n-dimensional hypercubes and make the circuit properties slightly more restrictive than required by our Theorem 1, then we succeed in showing that such circuits cannot exist. Thus, we come close to showing that the CFI-query over unordered hypercubes is undefinable by any CPT-algorithm from the class we are considering.

Following [9], we denote a CFI-graph over a base graph G or H as $\mathfrak{G}^S$ or $\mathfrak{H}^S$ (where S denotes the set of vertices whose CFI-gadget is odd). We consider circuits whose internal gates are XOR-gates and whose input gates are labelled with edges of the associated CFI base graph G. Therefore, the automorphism group of G has a natural action on the circuits as well. A circuit is said to be *sensitive* to a certain input bit if flipping just that bit changes the output. The other circuit properties will be explained in Section 5.

▶ **Theorem 1** (Main Theorem; see Theorem 31 in the full version for more details)**.** *Let $(G_n = (V_n, E_n))_{n\in\mathbb{N}}$ be a sequence of connected base graphs. Let $\mathfrak{G}_n^S$ be a CFI-graph over G_n, and let $\boldsymbol{tw}_n$ denote the treewidth of G_n. If there exists a CPT-program Π that is* super-symmetric *and* CFI-symmetric *and decides the CFI-query on all $\mathfrak{G}_n^S$, then there also exists a family $(C_n)_{n\in\mathbb{N}}$ of XOR-circuits such that*

1. *The number of gates in C_n is polynomial in $|\mathfrak{G}_n^S|$.*
2. *The $\mathbf{Aut}(G_n)$-orbit of the circuit has size polynomial in $|\mathfrak{G}_n^S|$.*
3. *C_n is* sensitive *to $\Omega(\boldsymbol{tw}_n)$ input bits.*
4. *The* fan-in dimension *of C_n is $\mathcal{O}(\log|\mathfrak{G}_n^S|)$.*

The terms *super-symmetric* and *CFI-symmetric* refer to the properties of a *hereditarily finite set* that is constructed by the program Π in order to decide the CFI-query. *Super-symmetry* is a property of h.f. sets that goes back to [9] (see Definition 5). *CFI-symmetry* is a concept that we define in this paper and which describes the internal structure and "local symmetries" of a h.f. set (see Section 4). The CFI-algorithms from [9] and [29] are based on super-symmetric and CFI-symmetric h.f. sets, and arguably, both these properties are crucial for the success of all these algorithms.

Our second main result shows that if we choose the n-dimensional hypercubes as the family of base graphs, and impose slightly stronger conditions on the circuits, then it is not possible to satisfy all of them together.

▶ **Theorem 2.** *Let $(\mathcal{H}_n)_{n\in\mathbb{N}}$ be the family of n-dimensional hypercubes and let $\boldsymbol{tw}_n$ denote the treewidth of $\mathcal{H}_n$. There exists no family of symmetric XOR-circuits $(C_n)_{n\in\mathbb{N}}$ such that:*

1. *The number of gates in C_n is polynomial in $|\mathfrak{H}_n^S|$.*
2. *The $\mathbf{Aut}(\mathcal{H}_n)$-orbit of the circuit has size exactly one.*
3. *C_n is* sensitive *to $\Omega(\boldsymbol{tw}_n)$ input bits.*
4. *For any two gates g, h in C_n such that h is a parent of g, it holds $|\mathbf{Orbit}_{(h)}(g)| \in \mathcal{O}(\log|\mathfrak{H}_n^S|)$ and $|\mathbf{Orbit}_{(g)}(h)| \in \mathcal{O}(\log|\mathfrak{H}_n^S|)$.*

Here, $\mathbf{Orbit}_{(h)}(g)$ denotes the orbit of the gate g with respect to the subgroup of $\mathbf{Aut}(\mathcal{H}_n)$ that fixes the gate h (and vice versa for $\mathbf{Orbit}_{(g)}(h)$).

If the four circuit properties were the same as in Theorem 1, then this would separate the class of super- and CFI-symmetric choiceless algorithms from PTIME. The difference between the two theorems is that here, the circuit has orbit size one, i.e. it is stabilised by the whole group $\mathbf{Aut}(\mathcal{H}_n)$, whereas in Theorem 1, the orbit of the circuit is only required to be polynomial. Moreover, here, we have a logarithmic bound on the parents and children (per orbit) of every gate, whereas in Theorem 1, the logarithmic bound is on the *fan-in dimension* of the gates. We define this notion in Section 5; we do not know if logarithmic fan-in dimension implies the orbit-wise logarithmic bound on the number of children (or vice versa), and probably, it does not imply the bound on the number of parents. So the gap between our two main results concerns how symmetric the circuits have to be and how restricted the connectivity between two consecutive circuit layers is. It remains a problem for future work to close this gap. This paper is thus a first step of a potentially longer programme towards showing CPT $\neq$ PTIME via circuit lower bounds.

Related work. Lower bounds for symmetric circuits are studied in different contexts in the literature. For example, families of highly symmetric Boolean circuits with threshold gates characterise the power of fixed-point logic with counting [1], and certain more general circuits capture rank logic [11]. Our results are different in the sense that the XOR-circuits do not *characterise* CPT; they only represent the relevant structure of the h.f. sets that CPT uses to decide the CFI-query. Another line of research focuses on lower bounds for symmetric arithmetic circuits for the determinant and permanent polynomials, with the aim of making progress towards separating VP from VNP [10, 12]. Other examples of symmetric circuit lower bounds concern Boolean circuits for the parity function [31] and for computing products of permutation matrices [21]. The technical methods we employ in the proof of Theorem 2 might be applicable to the study of symmetric circuits in other contexts as well but we have not investigated this yet.

2 Choiceless Polynomial Time

Hereditarily finite sets. Let A be a finite set of atoms. The set of hereditarily finite objects over A, $\mathrm{HF}(A)$, is defined as $\bigcup_{i\in\mathbb{N}} \mathrm{HF}_i(A)$, where $\mathrm{HF}_0(A) := A \cup \{\emptyset\}, \mathrm{HF}_{i+1}(A) := \mathrm{HF}_i(A) \cup 2^{\mathrm{HF}_i(A)}$. The size of a h.f. set $x \in \mathrm{HF}(a)$ is measured in terms of its *transitive closure* $\mathrm{tc}(x)$: The set $\mathrm{tc}(x)$ is the least transitive set such that $x \in \mathrm{tc}(x)$. Transitivity means that for every $a \in \mathrm{tc}(x)$, $a \subseteq \mathrm{tc}(x)$. Intuitively, one can view $\mathrm{tc}(x)$ as the set of all sets that appear as elements at some nesting depth within x.

Choiceless Polynomial Time. By CPT we always mean Choiceless Polynomial Time *with counting*. For details and various ways to define CPT formally, we refer to the literature: A concise survey can be found in [17]. The work by Blass, Gurevich and Shelah in which CPT was originally introduced as an abstract state machine model is [3]; later, more "logic-like" presentations of CPT were invented, such as Polynomial Interpretation Logic (see [16, 32]) and BGS-logic [30]. In short, CPT is like fixed-point logic with counting [6] plus a mechanism to construct isomorphism-invariant hereditarily finite sets of polynomial size. When a CPT-sentence (also called program) Π is evaluated in a finite structure $\mathfrak{A}$, then Π may augment $\mathfrak{A}$ with hereditarily finite sets over its universe. The total number of distinct sets appearing in them (i.e. the sum over the sizes of the transitive closures of the h.f. sets) and the number of computation steps is bounded by $p(|A|)$, where $p(n)$ is a polynomial that is explicitly part of the sentence Π. The run of Π on $\mathfrak{A}$ is formally a sequence of computation stages, each of which is a h.f. set.

The h.f. sets that appear in the run of a program Π on a structure $\mathfrak{A}$ are called the sets that are *activated* by Π on input $\mathfrak{A}$. Formally, the set of active objects is the union over the transitive closures of all the computation stages of the run of Π on $\mathfrak{A}$. The precise definition is not important for the purposes of this article and there exist multiple slightly varying definitions in the literature [9, 30, 32] which all essentially describe the same concept. The main limitation of CPT is that the set of objects activated by Π on $\mathfrak{A}$ is closed under the automorphisms of $\mathfrak{A}$ and at the same time of polynomial size in $|A|$. Thus, objects with super-polynomially large orbits cannot be activated by a CPT-program. It may be worth noting that while the whole set of activated objects has an orbit of size one, each activated object itself need not be fixed by *every* automorphism – as long as its orbit is only polynomial.

3 Unordered Cai-Fürer-Immerman graphs

Fix an undirected connected graph $G = (V, E)$ as the *base graph* for the CFI-construction (whenever we speak of base graphs throughout the paper, we mean connected graphs). We turn G into a CFI-graph by replacing the edges with certain edge-gadgets and the vertices with vertex-gadgets. There are two types of vertex-gadgets, called odd and even. To construct a concrete CFI-graph over G, we have to fix a set $S \subseteq V$ of vertices which are replaced by the *odd* gadget. The vertices in $V \setminus S$ will be turned into the *even* gadget. We denote the resulting CFI-graph by $\mathfrak{G}^S$. The precise definition is as follows: Let $\widehat{E} := \{e_0, e_1 \mid e \in E\}$. These are the vertices that will form the edge-gadgets of $\mathfrak{G}^S$, so there are two vertices per edge-gadget. To define the vertices in vertex-gadgets, we let, for each $v \in V$, $v^*_S := \{v^X \mid X \subseteq E(v), |X| \text{ even }\}$ if $v \notin S$, and otherwise, $v^*_S := \{v^X \mid X \subseteq E(v), |X| \text{ odd }\}$. Here, $E(v) \subseteq E$ are the edges incident to v in G. The vertices in v^*_S form the vertex-gadget of v. In total, we let $\widehat{V}_S := \bigcup_{v \in V} v^*_S$. Then the vertex-set of $\mathfrak{G}^S$ is $V(\mathfrak{G}^S) := \widehat{V}_S \cup \widehat{E}$. The edges of the CFI-graph are given by

$$E(\mathfrak{G}^S) := \{\{v^X, e_i\} \mid v^X \in \widehat{V}_S, e_i \in \widehat{E}, |X \cap \{e\}| = i\} \cup \{\{e_0, e_1\} \mid e \in E\}.$$

In other words, for every $v \in V$, we connect each $v^X \in v^*_S$ with the edge-gadgets of all edges $e \in E(v)$ in such a way that v^X is connected with e_0 if $e \notin X$, and otherwise with e_1. Also, we connect e_0 and e_1 to ensure that no automorphism of $\mathfrak{G}^S$ can tear apart the edge-gadgets. Our CFI-graphs are unordered, so the only relation of the structure $\mathfrak{G}^S$ is the edge relation E. The *CFI-query* asks for the parity of $|S|$, given a CFI-graph $\mathfrak{G}^S$. This is essentially the

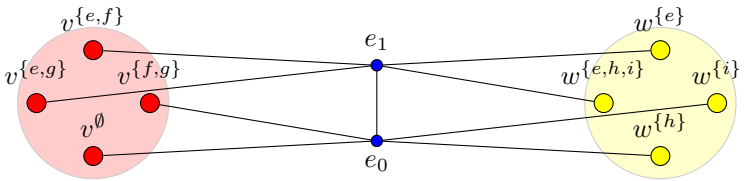

Figure 1 Gadgets v^*_S, w^*_S for $v \notin S, w \in S$, connected by the gadget for the edge e.

same question as the graph isomorphism problem for CFI-graphs:

▶ **Theorem 3** ([4, 9]). *For two given CFI-graphs over the same base graph, it holds* $\mathfrak{G}^S \cong \mathfrak{G}^R$ *if and only if* $|S| \equiv |R| \mod 2$.

Alternatively, deciding the parity of $|S|$ can be phrased as a linear equation system over $\mathbb{F}_2$ in the variables $\widehat{E}$ [2]. Since the reduction to a linear equation system is easily computable from the given CFI-graph $\mathfrak{G}^S$, and linear equation systems can be efficiently solved using, for example, Gaussian elimination, the CFI-query is decidable in polynomial time.

For logics that lack the ability to create higher-order objects, such as bounded-variable counting logic $\mathcal{C}^k$ (and hence fixed-point logic with counting), it is provably impossible to distinguish non-isomorphic CFI-graphs, provided that the treewidth of the base graphs is super-constant:

▶ **Theorem 4** ([4, 2]). *Let $G = (V, E)$ be an undirected connected graph with treewidth t. Then for any two sets $S, S' \subseteq V$, it holds $\mathfrak{G}^S \equiv_{\mathcal{C}^t} \mathfrak{G}^{S'}$, even if $\mathfrak{G}^S \ncong \mathfrak{G}^{S'}$.*

3.1 Automorphisms of unordered CFI-graphs

For a CFI-graph $\mathfrak{G}^S$ over an *unordered* base graph $G = (V, E)$, two different kinds of automorphisms play a role: Firstly, there are what we call "CFI-automorphisms". These are induced by swapping e_0 and e_1 in some edge-gadgets (this is called "flipping the edge"). Secondly, there are the automorphisms of the underlying graph G itself.

To speak about the CFI-automorphisms, we use the terminology from [9]: For a given base graph G, we consider not only a concrete CFI-instance with odd and even vertex gadgets, but we can also construct the "full" CFI-graph $\mathfrak{G}$, in which every vertex gadget is both even and odd. Formally, for $v \in V$, let $v^* := v^*_\emptyset \cup v^*_{\{v\}} = \{v^X \mid X \subseteq E(v)\}$, and $\widehat{V} := \bigcup_{v \in V} v^*$. The vertex-set of $\mathfrak{G}$ is $\widehat{V} \cup \widehat{E}$, and the edge-set is

$$E(\mathfrak{G}) := \{\{v^X, e_i\} \mid v^X \in \widehat{V}, e_i \in \widehat{E}, |X \cap \{e\}| = i\} \cup \{\{e_0, e_1\} \mid e \in E\}.$$

Every CFI-instance $\mathfrak{G}^S$ is an induced subgraph of $\mathfrak{G}$. Some of the CFI-automorphisms of $\mathfrak{G}$ are also automorphisms of $\mathfrak{G}^S$, but not all of them are. The other automorphisms of $\mathfrak{G}$ induce isomorphisms from $\mathfrak{G}^S$ into another isomorphic CFI-graph. For each edge $e = \{v, w\} \in E$, let ρ_e denote the automorphism of $\mathfrak{G}$ induced by flipping the edge e. Formally, $\rho_e(e_0) = e_1, \rho_e(e_1) = e_0$, and $\rho_e(v^X) = v^{X \triangle \{e\}}$, $\rho_e(w^X) = w^{X \triangle \{e\}}$ for all $v^X, w^X \in v^* \cup w^*$. All other vertices in $\widehat{V}$ are fixed by ρ_e. One can check that this is indeed an automorphism of $\mathfrak{G}$; furthermore, ρ_e is an *isomorphism* from any CFI-instance $\mathfrak{G}^S$ to $\mathfrak{G}^{S \triangle \{v,w\}}$ (see also [9]). It is easy to see that these edge-flip automorphisms commute, so for $F = \{e^1, ..., e^m\} \subseteq E$ we may write ρ_F for $\rho_{e^1} \circ \rho_{e^2} \circ ... \circ \rho_{e^m}$. So in total, for every $F \subseteq E$, ρ_F is an automorphism of $\mathfrak{G}$. If every $v \in V$ is incident to an even number of edges in F, then ρ_F is also an automorphism of $\mathfrak{G}^S$, not only of $\mathfrak{G}$. To sum up, we have the following groups of CFI-automorphisms of $\mathfrak{G}$ and $\mathfrak{G}^S$: $\mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G}) := \{\rho_F \mid F \subseteq E\}$. This group is isomorphic to the Boolean vector space $\mathbb{F}_2^E$: Each $F \subseteq E$ is identified with its characteristic vector $\chi(F) \in \mathbb{F}_2^E$. It holds $\rho_F \circ \rho_{F'} = \rho_{F \triangle F'}$, and this corresponds to the vector $\chi(F) + \chi(F') \in \mathbb{F}_2^E$. As already said, for a CFI-instance $\mathfrak{G}^S$, i.e. an induced subgraph of $\mathfrak{G}$, we have that $\mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G}^S)$ is isomorphic to a subspace of $\mathbb{F}_2^E$. In addition to the CFI-automorphisms, we also have to consider $\mathbf{Aut}(G) \leq \mathbf{Sym}(V)$, i.e. the automorphism group of the unordered base graph. In total, the automorphism group of the full CFI-graph $\mathfrak{G}$ is isomorphic to the following semi-direct product: $\mathbf{Aut}(\mathfrak{G}) \cong \mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G}) \rtimes \mathbf{Aut}(G) = \{(\rho_F, \pi) \mid \rho_F \in \mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G}), \pi \in \mathbf{Aut}(G)\}$. The automorphism group $\mathbf{Aut}(\mathfrak{G}^S)$ of a concrete CFI-instance is $\mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G}^S) \rtimes \mathbf{Aut}(G) \leq \mathbf{Aut}(\mathfrak{G})$.
Sets that are CPT-definable in $\mathfrak{G}^S$ only have to be symmetric with respect to the latter, but nonetheless, we also consider the full group $\mathbf{Aut}(\mathfrak{G})$ because it simplifies the analysis. The sets we call *super-symmetric* (see below) are also $\mathbf{Aut}(\mathfrak{G})$-symmetric.

3.2 Symmetries and supports of hereditarily finite sets over CFI-graphs

Let $\mathfrak{G}^S$ be a CFI-graph over $G = (V, E)$ and $x \in \mathrm{HF}(\widehat{E})$. All the groups from the previous section act on $\widehat{E}$ and therefore also on $\mathrm{HF}(\widehat{E})$. The action of any permutation π on a set $x \in \mathrm{HF}(\widehat{E})$ is given by $\pi(x) = \{\pi(y) \mid y \in x\}$. If x is an atom e_i, with $i \in \{0, 1\}$,

$e = \{u, v\} \in E$, and $\pi \in \mathbf{Aut}(G)$, then $\pi(x) = \pi(e)_i$, where $\pi(e) = \{\pi(u), \pi(v)\} \in E$. If $\pi = \rho_F \in \mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$, then $\pi(e_i) = e_j$, where $j = i + |F \cap \{e\}| \mod 2$. A permutation π *stabilises* an object $x \in \mathrm{HF}(\widehat{E})$, if $\pi(x) = x$. As already said, $\mathbf{Aut}(\mathfrak{G}^S)$ is composed of edge flips and automorphisms of the base graph. We separate the effect of these two subgroups on the elements of $\mathrm{HF}(\widehat{E})$ and consider the following orbits and stabilisers for $x \in \mathrm{HF}(\widehat{E})$. The different orbits we consider are $\mathbf{Orb}_E(x) := \{\rho_F(x) \mid \rho_F \in \mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})\}$, $\mathbf{Orb}_G(x) := \{\pi(x) \mid \pi \in \mathbf{Aut}(G)\}$, and the corresponding stabilisers are $\mathbf{Stab}_E(x) = \{\chi(F) \mid \rho_F \in \mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G}), \rho_F(x) = x\}$ and $\mathbf{Stab}_G(x) = \{\pi \in \mathbf{Aut}(G) \mid \pi(x) = x\}$. We always view $\mathbf{Stab}_E(x)$ as a subspace of the Boolean vector space $\mathbb{F}_2^E$. Furthermore, let $\mathbf{maxOrb}_E(x) := \max_{y \in \mathrm{tc}(x)} |\mathbf{Orb}_E(y)|$.

In [9], the term *super-symmetry* was introduced for h.f. sets which are fixed by *all* automorphisms in $\mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$. Here, we use a slightly relaxed notion:

▶ **Definition 5** (Super-symmetric objects). *Fix a family of CFI-graphs $(\mathfrak{G}_n^S)_{n\in\mathbb{N}}$ and a $\mu_n \in \mathrm{HF}(\widehat{E}_n)$ for every n. The objects μ_n are* super-symmetric *if there exists a polynomial p such that $|\mathbf{Orb}_E(\mu_n)| \leq p(|\mathfrak{G}_n^S|)$.*

Supports for CFI-automorphisms

Generally, a *support* of a permutation group $\Gamma \leq \mathbf{Sym}(A)$ is a subset $S \subseteq A$ such that the pointwise stabiliser of S in $\mathbf{Sym}(A)$ is a subgroup of Γ. A support of a h.f. set is a support of its stabiliser group. For subgroups of $\mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$, we will use a different notion, that we call *CFI-support*. The reason why we need a specific type of support for these groups is because otherwise, the group $\mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$ does not admit unique minimum supports.

▶ **Definition 6** (CFI-support). *A* CFI-support *of an object $x \in \mathrm{HF}(\widehat{E})$ is a subset $S \subseteq E$ such that every $\rho_F \in \mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$ with $F \cap S = \emptyset$ fixes x.*

For the proof of the next lemma, we refer to the long version; it is not very difficult and similar to the proof of Lemma 26 in [3]. The lemma entails that every $x \in \mathrm{HF}(\widehat{E})$ has a unique smallest CFI-support.

▶ **Lemma 7.** *Let $x \in \mathrm{HF}(\widehat{E})$. Let $A_1, A_2 \subseteq E$ be CFI-supports of x. Then $A_1 \cap A_2$ is also a CFI-support of x.*

▶ **Definition 8** (Minimal CFI-support). *For $x \in \mathrm{HF}(\widehat{E})$, $\sup_{\mathrm{CFI}}(x) \subseteq E$ denotes the unique minimal subset of E that is a CFI-support of x.*

4 CFI-symmetric hereditarily finite sets and algorithms

The CFI-query is definable in CPT on instances that arise from linearly ordered base graphs, base graphs that come with a preorder with colour classes of logarithmic size, and base graphs of linear degree [9, 29]. All these CPT-algorithms depend on the construction of a particular super-symmetric h.f. set $\mu \in \mathrm{HF}(\widehat{E})$ that encodes the parity of $|S|$, given an instance $\mathfrak{G}^S$. We isolate another property of these h.f. sets, besides super-symmetry, which is responsible for their small orbit size and suitability for encoding parities. We call this *CFI-symmetry*. Intuitively, a set $\mu \in \mathrm{HF}(\widehat{E})$ is CFI-symmetric if its "building blocks" behave similarly as CFI-gadgets in CFI-graphs, in the sense that they are "flipped" whenever an even number of "incident gadgets" is flipped. These building blocks are the *connected components* of sets. To define these, let a CFI-graph $\mathfrak{G}^S$ and a set $\mu \in \mathrm{HF}(\widehat{E})$ be fixed, and let $\sim_E$ be the following equivalence relation on the elements $x \in \mathrm{tc}(\mu)$: For $x, x' \in \mathrm{tc}(\mu)$, we write $x \sim_E x'$

iff there exists an edge-flip $\rho_F \in \mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$ such that $x' = \rho_F(x)$. The $\sim_E$-equivalence class in $\mathrm{tc}(\mu)$ of an object $x \in \mathrm{tc}(\mu)$ is denoted $[x]$. The relation $\sim_E$ induces a partition $\mathcal{C}(x)$ on each $x \in \mathrm{tc}(\mu)$, namely $\mathcal{C}(x) := \{([y] \cap x) \mid y \in x\}$. In [9], the elements of $\mathcal{C}(x)$ are called the *connected components* of x. Now in a *CFI-symmetric* object, each connected component $\gamma \in \mathcal{C}(x)$, for each $x \in \mathrm{tc}(\mu)$, behaves like a CFI-gadget. That is, the component has exactly two images under $\mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$, namely itself and its "flip". Consider the following example of a small "parity-tracking" h.f. set that is constructed similarly as in the algorithms from [9] and [29].

▶ **Example 9.** Here is an example h.f. set $\mu_{\{e,f,g\}} \in \mathrm{HF}(\widehat{E})$ with $E = \{e, f, g\}$. It tracks the parity of edge-flips for the edges e, f, g. For better readability, the set is printed in a structured form, so the sets $\mu_{\{f,g\}}$ and $\widetilde{\mu}_{\{f,g\}}$ are shown in the level below. Each of the

$$\mu_{\{e,f,g\}} = \Big\{ \{\mu_{\{f,g\}}, e_0\}, \{\widetilde{\mu}_{\{f,g\}}, e_1\} \Big\}$$

$$\{\{f_0, g_0\}, \{f_1, g_1\}\} \qquad \{\{f_0, g_1\}, \{f_1, g_0\}\}$$

μ-objects has only one connected component that consists of two sets which are related by $\sim_E$. For example, the set $\mu_{\{e,f,g\}}$ is stabilised setwise whenever an even number of edges is flipped. The two elements of $\mu_{\{e,f,g\}}$ themselves have two connected components: Clearly, e_0 and $\mu_{\{f,g\}}$ cannot be mapped to each other by any edge-flip. The same goes for example for f_0 and g_0. They form distinct components of the set $\{f_0, g_0\}$, while $\{\{f_0, g_0\}, \{f_1, g_1\}\}$ again only has one component that is stabilised if and only if an even number of edges in $\{f, g\}$ is flipped. This pattern of alternation between sets with two components and sets with one component is typical of the super-symmetric objects constructed by the known CFI-algorithms.

▶ **Definition 10** (CFI-symmetric components and objects). *Let $\mu \in \mathrm{HF}(\widehat{E})$, $x \in \mathrm{tc}(\mu)$, and $\gamma \subseteq x$ be a connected component of x. Then we say that γ is* CFI-symmetric *if $|\mathbf{Orb}_E(\gamma)| = 2$, and for each $\rho_F \in \mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$, it holds that $\rho_F(\gamma) = \gamma$ iff for one/every $y \in \gamma$, the number of flipped components of y, that is $|\{\gamma' \in \mathcal{C}(y) \mid \rho_F(\gamma') \neq \gamma'\}|$, is even.*

The set μ is CFI-symmetric *if the following two conditions are satisfied:*

1. *For each $\rho_F \in \mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$, it holds that $\rho_F(\mu) = \mu$ iff the number of flipped components of μ, that is, $|\{\gamma \in \mathcal{C}(\mu) \mid \rho_F(\gamma) \neq \gamma\}|$, is even.*
2. *For every $x \in \mathrm{tc}(\mu)$, every connected component $\gamma \in \mathcal{C}(x)$ is CFI-symmetric.*

In the full version, we show that the formulation "one/every" in the above definition is indeed justified. We call a CPT-program Π that decides the CFI-query on a class $\mathcal{K}$ of base graphs *CFI-symmetric* if it activates a CFI-symmetric h.f. set $\mu \in \mathrm{HF}(\widehat{E})$ on every input $\mathfrak{G}^S$ over every base graph $G \in \mathcal{K}$. To be precise, μ must also have sufficient support size in order to enable the program to decide the CFI-query. This support lower bound is stated in Theorem 16. Similarly, we say that Π is *super-symmetric* if it activates a super-symmetric object of sufficient support.

5 Translating hereditarily finite sets to XOR-circuits

An XOR-circuit is a connected directed acyclic graph $C = (V_C, E_C)$ with a unique designated root r. Its internal nodes are understood as XOR-gates and its leaves correspond to the input gates of the circuit. If $(g, h) \in E_C$, then the output of gate h is an input of gate g. Every XOR-circuit computes the Boolean XOR-function over a subset of its input bits.

We say that an XOR-circuit C is a circuit *over a graph* $G = (V, E)$, if the input gates of C are labelled with the edges in E. More precisely, let $L \subseteq V_C$ be the leaves of C. There is an injective labelling function $\ell : L \longrightarrow E$ that relates the input gates with edges of G. To speak about the semantics of the circuit, we introduce a set of formal propositional variables $\mathcal{V}(G) := \{X_e \mid e \in E\}$. Every input gate $g \in L$ is associated with the formal variable $X_{\ell(g)}$. Since every internal gate is an XOR-gate, the function computed by it is the XOR over a subset of $\mathcal{V}(G)$. For our purposes, this variable set (or actually the set of associated edges) is the main interesting property of a gate, and we call it $\mathcal{X}(g)$. Formally, if $g \in L$, then $\mathcal{X}(g) := \{\ell(g)\} \subseteq E$. If g is an internal gate, then $\mathcal{X}(g) := \bigtriangleup_{h \in gE_C} \mathcal{X}(h)$, that is, the symmetric difference over the $\mathcal{X}(h)$ for all children of g. In other words, $\mathcal{X}(g) \subseteq E$ is precisely the set of edges in E such that g computes the Boolean function $\bigoplus_{e \in \mathcal{X}(g)} X_e$. Thus, a gate g is *sensitive* to an input bit X_e if and only if $e \in \mathcal{X}(g)$. The function computed by the circuit C is the XOR over $\mathcal{X}(r) \subseteq E$, where r is the root of C.

5.1 Symmetries of circuits

A circuit C over a graph G is subject to the action of the automorphism group $\mathbf{Aut}(G) \leq \mathbf{Sym}(V)$. Any $\pi \in \mathbf{Aut}(G)$ changes the labels of the input gates in L. So let $g \in L$ with $\ell(g) = e \in E$. Then $\pi(g)$ is an input gate with $\ell(\pi(g)) = \pi(e)$. The circuit $\pi(C)$ is just C with the input labels modified accordingly. We say that π *extends to an automorphism* of C if there exists a bijection $\sigma : V_C \longrightarrow V_C$ that is an automorphism of the graph (V_C, E_C) and satisfies for each input gate $g \in V_C$: $\ell(\sigma(g)) = \pi(\ell(g))$. We write $\mathbf{Stab}_G(C) = \{\pi \in \mathbf{Aut}(G) \mid \pi \text{ extends to an automorphism of } C\} \leq \mathbf{Aut}(G)$, and $\mathbf{Orb}_G(C) = \{\pi(C) \mid \pi \in \mathbf{Aut}(G)\}$.

5.2 The parameter fan-in dimension

The XOR-circuits we will construct from CFI-symmetric h.f. sets will satisfy a certain fan-in bound on the gates. However, this bound will not be – as it is more common – on the number of incoming wires of a gate but rather, on the "linear algebraic complexity of incoming information", so to say. The subsets of E form a Boolean vector space together with the symmetric difference operation. This space is isomorphic to $\mathbb{F}_2^E$.

For each gate g of an XOR-circuit $C = (V_C, E_C)$, gE_C denotes the set of its children and $E_C g$ the set of parents. With each internal gate g of an XOR-circuit C over a graph $G = (V, E)$, we can associate a Boolean matrix $M(g) \in \mathbb{F}_2^{gE_C \times E}$, that we call the *gate matrix*: The row at index $h \in gE_C$ is defined as the characteristic vector of $\mathcal{X}(h) \subseteq E$, transposed, i.e. $M(g)_{h-} = \chi(\mathcal{X}(h))^T$. Here and in what follows, we write χ for the bijection from $\mathcal{P}(E)$ to $\mathbb{F}_2^E$ that associates with each subset of E its characteristic Boolean vector. If g is an input gate, then we define $M(g) \in \mathbb{F}_2^{[1] \times E}$ as the one-row matrix whose only row is $\chi(\mathcal{X}(g)))^T = \chi(\{\ell(g)\})^T$.

The *fan-in dimension* of a gate g is the dimension of the row-space of $M(g)$; this is the subspace of $\mathbb{F}_2^E$ that is spanned by the characteristic vectors $\chi(\mathcal{X}(h)) \in \mathbb{F}_2^E$, for all children h of g. Equivalently, the fan-in dimension of g is $\mathbf{rk}(M(g))$. The *fan-in dimension* of the circuit C is the maximum fan-in dimension of any of its gates. The notion of fan-in dimension is unusual but as we will show, it nicely captures the orbit size of the original h.f. set with respect to the group of edge flips $\mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$. The $\mathbf{Aut}(G)$-symmetries of the h.f. set will be reflected in the symmetries of the circuit.

5.3 The circuit construction

▶ **Theorem 11.** *Fix a family $(G_n)_{n\in\mathbb{N}}$ of base graphs. For every $n \in \mathbb{N}$, let $\mathfrak{G}_n^S$ be a CFI-graph over $G_n = (V_n, E_n)$ and let $\mu_n \in \mathrm{HF}(\widehat{E}_n)$ be a* CFI-symmetric *h.f. set that is activated by a CPT-program on input $\mathfrak{G}_n^S$ (by the same CPT-program for the whole family of graphs). Then for every $n \in \mathbb{N}$, there exists an XOR-circuit $C(\mu_n) = (V_C, E_C)$ over the edges of G_n which satisfies:*

1. *The size of the circuit, i.e. $|V_C|$, is polynomial in $|\mathfrak{G}_n^S|$.*
2. *The orbit-size $|\mathbf{Orb}_G(C(\mu_n))|$ of the circuit is polynomial in $|\mathfrak{G}_n^S|$.*
3. *$C(\mu_n)$ is* sensitive *to an edge $e \in E_n$ if and only if $e \in \sup_{\mathrm{CFI}}(\mu_n)$.*
4. *The* fan-in dimension *of $C(\mu)$ is $\mathcal{O}(\log(\mathbf{maxOrb}_E(\mu)))$. Recall that $\mathbf{maxOrb}_E(\mu) = \max_{y\in\mathrm{tc}(x)} |\mathbf{Orb}_E(y)|$.*

For the proof, we fix $\mu \in \mathrm{HF}(\widehat{E})$ and denote by $C(\mu) = (V_C, E_C)$ the corresponding XOR-circuit that we are going to define. The circuit is simply the factorised DAG-structure $(\mathrm{tc}(\mu), \in)_{/\sim_E}$: The gates of the circuit are the $\sim_E$-equivalence classes (i.e. $\mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$-orbits) of the objects in $\mathrm{tc}(\mu)$. Note that these orbits need not be subsets of $\mathrm{tc}(\mu)$, so whenever we write $[x]$, we formally mean the orbit restricted to $\mathrm{tc}(\mu)$: $[x] = \{\rho_F(x) \mid \rho_F \in \mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$ such that $\rho_F(x) \in \mathrm{tc}(\mu)\}$. The circuit $C(\mu)$ is defined as follows:

- $V_C := \mathrm{tc}(\mu)_{/\sim_E} = \{[x] \mid x \in \mathrm{tc}(\mu)\}$.
- $E_C := \{([x], [y]) \mid$ there exists $y' \in [y]$ such that $y' \in x\}$.
- By definition, the leaves of $C(\mu)$ correspond to $\sim_E$-classes of atoms in $\mathrm{tc}(\mu)$. The set of atoms is $\widehat{E}$, so any leaf of C has the form $[e_i]$, for some $e \in E, i \in \{0, 1\}$. We let $\ell([e_i]) := e$.
- The root r of $C(\mu)$ is $[\mu]$.

One can prove that the set of edges E_C can indeed be defined in this way: Whether or not there is an E_C-edge between $[x]$ and $[y]$ is independent of the choice of the representative of $[x]$ in the definition. This is because all members of $[x]$ are symmetric to each other in the DAG-structure $(\mathrm{tc}(\mu), \in)$. Now we verify that the circuit has the desired properties. We start with the $\mathbf{Aut}(G)$-symmetry.

▶ **Lemma 12.** *Every $\pi \in \mathbf{Stab}_G(\mu) \leq \mathbf{Sym}(V)$ extends to an automorphism of the circuit $C(\mu)$, that is: $\mathbf{Stab}_G(\mu) \leq \mathbf{Stab}_G(C(\mu))$.*

Proof sketch. Let $\pi \in \mathbf{Stab}_G(\mu) \leq \mathbf{Aut}(G)$. That is, π extends to an automorphism $\sigma : \mathrm{tc}(\mu) \longrightarrow \mathrm{tc}(\mu)$ of $(\mathrm{tc}(\mu), \in)$. We define $\sigma' : V_C \longrightarrow V_C$ by letting $\sigma'([x]) = [\sigma(x)]$. This is well-defined because $x \sim_E x'$ if and only if $\sigma(x) \sim_E \sigma(x')$ (σ is an automorphism of μ). One can verify that σ' is an automorphism of $C(\mu)$ that π extends to. ◀

▶ **Corollary 13.** $|\mathbf{Orb}_G(C(\mu))| \leq |\mathbf{Orb}_G(\mu)|$.

Proof. Follows from Lemma 12 together with the Orbit-Stabiliser Theorem, which says that $|\mathbf{Orb}_G(C(\mu))| = |\mathbf{Aut}(G)|/|\mathbf{Stab}_G(C(\mu))|$ and $|\mathbf{Orb}_G(\mu)| = |\mathbf{Aut}(G)|/|\mathbf{Stab}_G(\mu)|$. ◀

Next, we would like to analyse the *fan-in dimension* of $C(\mu)$, and the connection between $C(\mu)$ and $\sup_{\mathrm{CFI}}(\mu)$. The key for this is to establish a connection between the stabilisers $\mathbf{Stab}_E(x)$, for all $x \in \mathrm{tc}(\mu)$, and the kernels of the corresponding *gate matrices*. For the definition of these matrices, we refer back to Section 5.2.

▶ **Lemma 14.** *For every gate $[x] \in V_C$ and its gate matrix $M[x] \in \mathbb{F}_2^{[x]E_C \times E}$, it holds: $\boldsymbol{Ker}(M[x]) = \mathbf{Stab}_E(x) = \mathbf{Stab}_E(x')$ for every $x' \in [x]$.*
For every row $M[x]_{[y]-}$, for every $[y] \in [x]E_C$, it holds:

$$\boldsymbol{Ker}(M[x]_{[y]-}) = \mathbf{Stab}_E([y] \cap x) \qquad (\star)$$

Proof. It holds $\mathbf{Stab}_E(x) = \mathbf{Stab}_E(x')$, for every $x' \in [x]$ and also $\mathbf{Stab}_E([y] \cap x) = \mathbf{Stab}_E([y] \cap x')$, for every $x' \in [x]$ (because $\mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$ is Abelian). Therefore, equation $(\star)$ does not depend on the choice of representatives. From $(\star)$ it immediately follows that $\mathbf{Ker}(M[x]) = \mathbf{Stab}_E(x)$, because: The stabiliser of x is the intersection of the stabilisers of all connected components of x, and the kernel of $M[x]$ is the intersection of the kernels of the individual rows of the matrix. We now prove $(\star)$ via induction from the input gates to the root. If $[x] = [e_0]$ is an input gate, then $M[x]$ has just one row, which is $\chi(e)^T$. The kernel of $\chi(e)^T$ is the set of all vectors in $\mathbb{F}_2^E$ which are zero at index e. This is precisely $\mathbf{Stab}_E(e_0) = \mathbf{Stab}_E(e_1)$, as desired. Now suppose $[x]$ is an internal gate, i.e. x is a non-atomic h.f. set in $\mathrm{tc}(\mu)$. Each row of $M[x] \in \mathbb{F}_2^{[x]E_C \times E}$ is the characteristic vector of $\mathcal{X}[y] \subseteq E$, for a $[y] \in [x]E_C$. Now fix such a child y of x. We have $\mathcal{X}[y] = \bigtriangleup_{[w] \in [y]E_C} \mathcal{X}[w]$. In matrix-vector notation, we can write this as:

$$M[x]_{[y]-} = \chi(\mathcal{X}([y]))^T = \sum_{[w] \in [y]E_C} (M[y]_{[w]-})^T = (1\ 1\ \ldots\ 1) \cdot M[y].$$

Let $\gamma \in \mathcal{C}(x)$ be the connected component such that $\gamma = [y] \cap x$. The equation above means that $\mathbf{Ker}(M[x]_{[y]-}) = \mathcal{E}_y$, where $\mathcal{E}_y$ denotes the set of all vectors in $\mathbb{F}_2^E$ whose image under $M[y]$ has even Hamming weight. Thus we have to show that $\mathcal{E}_y = \mathbf{Stab}_E(\gamma)$. Each row $M[y]_{[w]-}$ corresponds to a connected component $\gamma' \in \mathcal{C}(y)$ with $w \in \gamma'$.
By the induction hypothesis, we have for each row $M[y]_{[w]-}$ and each $\mathbf{v} \in \mathbb{F}_2^E$ that $M[y]_{[w]-} \cdot \mathbf{v} = 1$ iff $\mathbf{v} \notin \mathbf{Stab}_E([w] \cap y)$. So $M[y] \cdot \mathbf{v}$ has even Hamming weight iff $\rho_{\chi^{-1}(\mathbf{v})} \in \mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$ flips an even number of connected components of y. This is true iff $\rho_{\chi^{-1}(\mathbf{v})}$ flips an even number of components in every $y' \in \gamma$. By definition of CFI-symmetry (Definition 10), this is the case iff $\mathbf{v} \in \mathbf{Stab}_E(\gamma)$, because μ is CFI-symmetric, and thus, γ is a CFI-symmetric component. In total, we have shown that $\mathbf{v} \in \mathcal{E}_y$ iff $\mathbf{v} \in \mathbf{Stab}_E(\gamma)$. This proves $(\star)$ for every row of $M[x]$. ◀

As a consequence of this correspondence between kernels and stabilisers, we can bound the fan-in dimension of $C(\mu)$. This proves **Property 4** from Theorem 11.

▶ **Lemma 15.** *The* fan-in dimension *of* $C(\mu)$ *is* $\log(\mathbf{maxOrb}_E(\mu))$.

Proof. Let $x \in \mathrm{tc}(\mu)$. From the Orbit-Stabiliser Theorem and the fact that $|\mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})| = 2^{|E|}$, it follows that $\mathbf{Orb}_E(x) = \frac{2^{|E|}}{|\mathbf{Stab}_E(x)|} \leq \mathbf{maxOrb}_E(\mu)$. By Lemma 14, $\mathbf{Stab}_E(x) = \mathbf{Ker}(M[x])$. Applying the Rank Theorem to $M[x]$, we get: $\mathbf{rk}(M[x]) = |E| - \dim \mathbf{Stab}_E(x) \leq \log(\mathbf{maxOrb}_E(\mu))$. Since there is an object $x \in \mathrm{tc}(\mu)$ where $\mathbf{maxOrb}_E(\mu)$ is attained, $\mathbf{rk}(M[x]) = \log(\mathbf{maxOrb}_E(\mu))$ is indeed the maximum rank of any gate matrix of $C(\mu)$. ◀

Proof of Theorem 11. First of all, since μ is by assumption activated by a CPT-sentence in the structure $\mathfrak{G}^S$, the size $|\mathrm{tc}(\mu)|$ and the orbit $|\mathbf{Orb}_{\mathbf{Aut}(\mathfrak{G}^S)}(\mu)|$ are polynomial in $|\mathfrak{G}^S|$. Therefore, **Property 1** from Theorem 11 clearly holds for $C(\mu)$, because $|V_C| \leq |\mathrm{tc}(\mu)|$. **Property 2** follows from the bound on $|\mathbf{Orb}_{\mathbf{Aut}(\mathfrak{G}^S)}(\mu)|$ together with Corollary 13, and the fact that $|\mathbf{Orb}_G(\mu)| \leq |\mathbf{Orb}_{\mathbf{Aut}(\mathfrak{G}^S)}(\mu)|$. **Property 4** is proven in Lemma 15. Finally, **Property 3** can be seen as follows: Suppose $C(\mu)$ is sensitive to an edge $e \in E$. This means that $e \in \mathcal{X}(r)$, for the root $r = [\mu]$ of $C(\mu)$. This is the case iff $e \in \mathcal{X}[y]$ for an odd number of children $[y] \in [\mu]E_C$. This is the same as saying that the column $M[\mu]_{-e}$ has odd Hamming weight. By equation $(\star)$ from Lemma 14, this holds if and only if $\chi(e) \notin \mathbf{Stab}_E([y] \cap x)$ for an odd number of children $[y] \in [\mu]E_C$. Since μ is CFI-symmetric, by Definition 10 this is the case if and only if $\rho_e(\mu) \neq \mu$. And this holds iff $e \in \mathrm{sup}_{\mathrm{CFI}}(\mu)$ (because $\mathrm{sup}_{\mathrm{CFI}}(\mu)$ is the smallest possible CFI-support of μ). ◀

5.4 Proving the main theorem

So far, we have a translation of CFI-symmetric h.f. sets in $\mathrm{HF}(\widehat{E})$ into XOR-circuits with the properties mentioned in Theorem 11. In order to conclude Theorem 1 from this, we additionally need the following: Any CPT-algorithm which is both super-symmetric and CFI-symmetric and decides the CFI-query must construct a h.f. set whose properties translate into the circuit properties from Theorem 1. Fortunately, a result to this effect exists already. The following support lower bound for general CPT-programs deciding the CFI-query is due to Dawar, Richerby, and Rossman [9].

▶ **Theorem 16** (implicit in the proof of Theorem 40 in [9]). *Let $(G_n)_{n\in\mathbb{N}}$ be a family of base graphs and let $\boldsymbol{tw}_n$ denote the treewidth of G_n. Let $\mathfrak{G}_n^0, \mathfrak{G}_n^1$ denote the even and odd CFI-structures over G_n. Assume that $\mathfrak{G}_n^0$ and $\mathfrak{G}_n^1$ are $\mathcal{C}^{\boldsymbol{tw}_n}$-homogeneous.*
Then any CPT-program that distinguishes $\mathfrak{G}_n^0$ and $\mathfrak{G}_n^1$ for all $n \in \mathbb{N}$ must activate on input $\mathfrak{G}_n^i$ a h.f. set μ_n whose smallest support *has size at least $\Omega(\boldsymbol{tw}_n)$.*

A structure $\mathfrak{G}_n^S$ is *$\mathcal{C}^{f(n)}$-homogeneous* if whenever two tuples $\overline{a}$ and $\overline{b}$ satisfy exactly the same $\mathcal{C}^{f(n)}$-formulas in $\mathfrak{G}_n^S$, then there is an automorphism of $\mathfrak{G}_n^S$ that maps $\overline{a}$ to $\overline{b}$. In particular, this condition is satisfied by certain ordered CFI-graphs, as stated in [9], and one can also show that the unordered CFI-graphs over hypercubes, which we use for the lower bound in Theorem 2, satisfy it. There are other details which must be taken into account when connecting Theorem 16 with Theorem 11 in order to prove Theorem 1, e.g. one has to reconcile the different notions of support that these theorems talk about. We gloss over these things in this extended abstract and refer to Theorem 31 in the long version, which is the more precise formulation of Theorem 1. For our application to hypercube CFI-structures, it is shown in the long version that Theorem 1 really holds in this shortened formulation. The reason why Theorem 1 requires the object to be *super-symmetric* is that this allows us to infer that $\mathcal{O}(\log \mathbf{maxOrb}_E(\mu)) \leq \mathcal{O}(\log |\mathfrak{G}_n^S|)$. This is needed to translate Property 4 from Theorem 11 into Property 4 from Theorem 1. Super-symmetry together with the fact that $|\mathrm{tc}(\mu)|$ is polynomial guarantees that $\mathbf{maxOrb}_E(\mu)$ is polynomially bounded in $|\mathfrak{G}_n^S|$ (see Definition 5).

6 A lower bound for symmetric XOR-circuits

The detailed proof of Theorem 2 is too long for this extended abstract but here is an outline: The theorem states that no family of XOR-circuits exists which are stabilised by all automorphisms of the n-dimensional hypercube, are sensitive to sufficiently many input bits, of polynomial size, and satisfy certain logarithmic bounds on the orbit-wise number of parents and children of each gate. What the proof concretely shows is that the sensitivity requirement (condition 3 in Theorem 2) contradicts the other three conditions: Any sufficiently symmetric circuit C is so highly connected that most input bits cancel themselves out because the number of distinct (not necessarily vertex-disjoint) paths from the root r to the input is even, and every operation is XOR. To prove that this self-cancellation effect takes place for an input gate g, we partition the set of paths between g and r in C into their orbits. Then we show that each orbit contains an even number of paths. To achieve this, it suffices to look at one path P in each orbit and to show the existence of an edge (h', h) in this path such that $\mathbf{Orbit}_{(h)}(h') := \{\pi(h') \mid \pi \in \mathbf{Stab}(h)\}$ has even size. Then we can conclude that P splits into an even number of "alternative routes" towards the root at gate h. Thus, the goal is to show that such an edge (h', h) exists on every path between r and g.

We do this by maintaining fine-grained information about the stabiliser groups $\mathbf{Stab}(h_i)$ of the gates $(h_1 = g, h_2, h_3, ..., r)$ along a given path (the path here is presented in the reverse direction of the edges). The automorphism group of the n-dimensional hypercube contains as a subgroup the symmetric group $\mathbf{Sym}_n$ acting on the n positions of the binary words $\{0,1\}^n$, which form the vertex set of the hypercube. Thus, each $\mathbf{Stab}(h_i)$ can be seen as a subgroup of $\mathbf{Sym}_n$. Such subgroups can be approximated by what we call their *coarsest alternating supporting partition* $\mathbf{SP}_A(\mathbf{Stab}(h_i))$ (inspired by a similar concept used in [1]). This is the coarsest partition $\mathcal{P}$ of $[n]$ such that for each $P \in \mathcal{P}$, every even permutation of P is a member of $\mathbf{Stab}(h_i)$. We know what $\mathbf{SP}_A(\mathbf{Stab}(g))$ looks like for every input gate g: This depends on the edge of the hypercube that g is labelled with. For a large proportion of the hypercube edges, this is a partition of $[n]$ into two parts of size $\Theta(n)$. We also know for the root that $\mathbf{SP}_A(\mathbf{Stab}(r)) = \{[n]\}$ because the circuit is stabilised by every permutation in $\mathbf{Sym}_n$. The bounds on the orbit-wise fan-in and fan-out degree that we assume in Theorem 2 allow us to prove that when we pass from any h_i to h_{i+1}, then $\mathbf{SP}_A(\mathbf{Stab}(h_i)) \approx \mathbf{SP}_A(\mathbf{Stab}(h_{i+1}))$, i.e. the supporting partitions hardly change (because if they did change more, then one would find that there must be more parent/child gates per orbit than allowed). Since $\mathbf{SP}_A(\mathbf{Stab}(g))$ differs quite substantially from $\mathbf{SP}_A(\mathbf{Stab}(r))$ but $\mathbf{SP}_A(\mathbf{Stab}(h_i)) \approx \mathbf{SP}_A(\mathbf{Stab}(h_{i+1}))$ for every i, we can infer that all "intermediate partitions" between $\mathbf{SP}_A(\mathbf{Stab}(g))$ and $\mathbf{SP}_A(\mathbf{Stab}(r))$ appear for the gate stabilisers along the path. Then we show that one of these "intermediate partitions" which must appear as $\mathbf{SP}_A(\mathbf{Stab}(h_i))$ for some i has properties which entail what we wanted, namely that $\mathbf{Orbit}_{(h_i)}(h_{i+1})$ has even size. Thus, the edge (h_{i+1}, h_i) on the path satisfies what we were looking for. So in short, the proof works because: Firstly, in order to show that a path has an even number of automorphic images, it suffices to find one gate on the path where the number of symmetric predecessors is even. Secondly, whether or not this happens can be inferred from the supporting partition of the gate. Thirdly, the fan-in and fan-out bounds enable us to track very precisely how the supporting partitions along the path look like. With these arguments, we can show for a sufficient number of input gates g, that they do not contribute to the result of the XOR-computation because they have an even number of paths to r – thus, circuit property 3 in Theorem 2 is not satisfied when the other conditions are. Carrying out this proof sketch requires quite some work that we have swept under the carpet here; for example, one needs certain group-theoretic arguments similar to the proof of Theorem 5.2 B in [13] in order to be able to describe the stabiliser groups appropriately with alternating supporting partitions.

7 Conclusion and future research

We have shown that the definability of the CFI-query on a class $\mathcal{K}$ of base graphs by means of a CFI-symmetric algorithm presupposes the existence of symmetric XOR-circuits with the properties from Theorem 1. We come close to proving the non-existence of these circuits for $\mathcal{K}$ being the family of n-dimensional hypercubes. It remains as a problem for future research to improve this lower bound and close the gap between the circuit properties in Theorems 1 and 2 in order to separate the class of CFI-symmetric algorithms from PTIME. Once that is achieved, the next step would be to lift this circuit approach to *all* CPT-algorithms for the CFI-query, not just the CFI-symmetric ones. This is a potential route to eventually solve the extremely difficult problem of separating CPT from PTIME. In the full version of the paper, we also provide a generalised circuit construction which works for a larger class of h.f. sets than CFI-symmetric ones. These are sets whose $\mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$-symmetries and $\mathbf{Aut}(G)$-symmetries

go well together in the sense that the Boolean vector spaces describing the $\mathbf{Aut}_{\mathrm{CFI}}(\mathfrak{G})$-stabilisers admit $\mathbf{Aut}(G)$-symmetric bases. However, one can prove that, unfortunately, the class of h.f. sets with this property is still not the full class of all CPT-definable sets.

A different approach towards CPT lower bounds could be to study the isomorphism problem of *multipedes* [25] instead of CFI-graphs. Such structures have been used to obtain lower bounds against individualization-refinement graph isomorphism algorithms. Multipedes differ from CFI-graphs in so far as they have no non-trivial automorphisms. Their inherent symmetries are rather given by counting-logic *types* (two tuples in a structure have the same *k-type* if they satisfy the same $\mathcal{C}^k$-formulas). It could be that the circuit construction is adaptable to this kind of symmetry. This would be of particular interest in case that the (unordered) CFI-query turns out to be in fact CPT-definable; then, multipedes might still provide an example to separate CPT from Ptime.

References

1 Matthew Anderson and Anuj Dawar. On symmetric circuits and fixed-point logics. *Theory of Computing Systems*, 60(3):521–551, 2017.

2 Albert Atserias, Andrei Bulatov, and Anuj Dawar. Affine systems of equations and counting infinitary logic. *Theoretical Computer Science*, 410(18):1666–1683, 2009.

3 Andreas Blass, Yuri Gurevich, and Saharon Shelah. Choiceless polynomial time. *Annals of Pure and Applied Logic*, 100(1-3):141–187, 1999.

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

5 Ashok Chandra and David Harel. Structure and complexity of relational queries. In *21st Annual Symposium on Foundations of Computer Science (sfcs 1980)*, pages 333–347. IEEE, 1980. doi:10.1109/SFCS.1980.41.

6 Anuj Dawar. The nature and power of fixed-point logic with counting. *ACM SIGLOG News*, 2(1):8–21, 2015.

7 Anuj Dawar, Erich Grädel, and Wied Pakusa. Approximations of Isomorphism and Logics with Linear-Algebraic Operators. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 112:1–112:14, 2019. doi:10.4230/LIPIcs.ICALP.2019.112.

8 Anuj Dawar, Erich Grädel, and Moritz Lichter. Limitations of the invertible-map equivalences. *Journal of Logic and Computation*, September 2022. doi:10.1093/logcom/exac058.

9 Anuj Dawar, David Richerby, and Benjamin Rossman. Choiceless Polynomial Time, Counting and the Cai–Fürer–Immerman graphs. *Annals of Pure and Applied Logic*, 152(1-3):31–50, 2008.

10 Anuj Dawar and Gregory Wilsenach. Symmetric Arithmetic Circuits. In *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*, volume 168 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 36:1–36:18, 2020. doi:10.4230/LIPIcs.ICALP.2020.36.

11 Anuj Dawar and Gregory Wilsenach. Symmetric circuits for rank logic. *ACM Transactions on Computational Logic (TOCL)*, 23(1):1–35, 2021.

12 Anuj Dawar and Gregory Wilsenach. Lower Bounds for Symmetric Circuits for the Determinant. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, volume 215 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 52:1–52:22, 2022. doi:10.4230/LIPIcs.ITCS.2022.52.

13 John Dixon and Brian Mortimer. *Permutation Groups*. Springer, New York, 1996.

14 Ronald Fagin. Generalized first-order spectra and polynomial-time recognizable sets. *Complexity of computation*, 7:43–73, 1974.

15 F. Gire and H.K. Hoang. An extension of fixpoint logic with a symmetry-based choice construct. *Information and Computation*, 144(1):40–65, 1998. doi:10.1006/inco.1998.2712.

16 E. Grädel, W. Pakusa, S. Schalthöfer, and L. Kaiser. Characterising Choiceless Polynomial Time with First-Order Interpretations. In *Proceedings of the 30th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 677–688, 2015.

17 Erich Grädel and Martin Grohe. Is Polynomial Time Choiceless? In *Fields of Logic and Computation II*, pages 193–209. Springer, 2015.

18 Martin Grohe. The quest for a logic capturing PTIME. In *2008 23rd Annual IEEE Symposium on Logic in Computer Science*, pages 267–271. IEEE, 2008. doi:10.1109/LICS.2008.11.

19 Martin Grohe, Pascal Schweitzer, and Daniel Wiebking. Deep Weisfeiler Leman, 2020. arXiv:2003.10935.

20 Yuri Gurevich. Logic and the Challenge of Computer Science. In *Current Trends in Theoretical Computer Science*. Computer Science Press, 1988.

21 William He and Benjamin Rossman. Symmetric formulas for products of permutations, 2022. arXiv:2211.15520.

22 Moritz Lichter. Separating Rank Logic from Polynomial Time. *J. ACM*, November 2022. doi:10.1145/3572918.

23 Moritz Lichter. Witnessed Symmetric Choice and Interpretations in Fixed-Point Logic with Counting, 2022. arXiv:2210.07869.

24 Moritz Lichter and Pascal Schweitzer. Choiceless Polynomial Time with Witnessed Symmetric Choice. In *LICS '22: 37th Annual ACM/IEEE Symposium on Logic in Computer Science, Haifa, Israel, August 2 - 5, 2022*, LICS '22. Association for Computing Machinery, 2022. doi:10.1145/3531130.3533348.

25 Daniel Neuen and Pascal Schweitzer. An exponential lower bound for individualization-refinement algorithms for graph isomorphism. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pages 138–150, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3188745.3188900.

26 Benedikt Pago. Choiceless Computation and Symmetry: Limitations of Definability. In *29th EACSL Annual Conference on Computer Science Logic (CSL 2021)*, volume 183 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 33:1–33:21, 2021. doi:10.4230/LIPIcs.CSL.2021.33.

27 Benedikt Pago. Lower bounds for Choiceless Polynomial Time via Symmetric XOR-circuits, 2023. arXiv:2302.05426.

28 Wied Pakusa. *Linear Equation Systems and the Search for a Logical Characterisation of Polynomial Time*. PhD thesis, RWTH Aachen, 2015.

29 Wied Pakusa, Svenja Schalthöfer, and Erkal Selman. Definability of Cai-Fürer-Immerman problems in Choiceless Polynomial Time. *ACM Transactions on Computational Logic (TOCL)*, 19(2):1–27, 2018. doi:10.1145/3154456.

30 Benjamin Rossman. Choiceless Computation and Symmetry. In *Fields of Logic and Computation*, pages 565–580. Springer, 2010.

31 Benjamin Rossman. Subspace-Invariant AC^0 Formulas. *Logical Methods in Computer Science*, 15, 2019.

32 Svenja Schalthöfer. *Choiceless Computation and Logic*. PhD thesis, RWTH Aachen, 2020.