Finite Model Theory and Proof Complexity Revisited: Distinguishing Graphs in Choiceless Polynomial Time and the Extended Polynomial Calculus

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

Abstract

This paper extends prior work on the connections between logics from finite model theory and propositional/algebraic proof systems. We show that if all non-isomorphic graphs in a given graph class can be distinguished in the logic *Choiceless Polynomial Time* with counting (CPT), then they can also be distinguished in the *bounded-degree extended polynomial calculus* (EPC), and the refutations have roughly the same size as the resource consumption of the CPT-sentence. This allows to transfer lower bounds for EPC to CPT and thus constitutes a new potential approach towards better understanding the limits of CPT. A super-polynomial EPC lower bound for a PTIME-instance of the graph isomorphism problem would separate CPT from PTIME and thus solve a major open question in finite model theory.
Further, using our result, we provide a model theoretic proof for the separation of bounded-degree polynomial calculus and bounded-degree *extended* polynomial calculus.

2012 ACM Subject Classification Theory of computation → Finite Model Theory

Keywords and phrases finite model theory, proof complexity, graph isomorphism

Digital Object Identifier 10.4230/LIPIcs.CSL.2023.31

Related Version *Full Version*: `https://arxiv.org/abs/2206.05086` [24]

Acknowledgements I would like to thank Daniel Wiebking for extensively answering my numerous questions on Deep Weisfeiler Leman and the properties of coherent configurations.

1 Introduction and results

In recent years, a close connection between propositional proof complexity and finite model theory has been discovered and investigated – this is fruitful in particular because it allows the transfer of lower bounds between the two fields. In [3], Berkholz and Grohe showed that, with respect to the *graph isomorphism problem*, *fixed-point logic with counting* (FPC) has the same expressive power as the *bounded-degree monomial calculus*. In [14] it was shown more generally that there are mutual simulations between different variants of fixed-point logic and resolution/monomial calculus, not only for the graph isomorphism problem, but for deciding any classes of finite structures. These simulations preserve the relevant complexity parameters: The number of variables in a fixed-point sentence is reflected in the width/degree of the corresponding resolution/monomial calculus refutation, and vice versa. Therefore, known lower bounds for these respective parameters can be transferred between proof complexity and finite model theory. We extend this line of research from the rather well-understood fixed-point logics to a stronger model of computation, *Choiceless Polynomial Time* (CPT). This logic is an extension of FPC with a mechanism to construct (isomorphism-invariant) higher-order objects, i.e. nested sets, over the input structure. This power to create new objects puts CPT in a realm beyond formalisms with bounded variable number, bounded width, or bounded degree. As it turns out, a proof system that can naturally simulate

this mechanism is the bounded-degree *polynomial calculus with extension axioms* over $\mathbb{Q}$. *Extension axioms* can be added to any proof system; they allow to introduce new variables in a proof as abbreviations for more complex expressions, which generally allows for considerably shorter proofs.

We consider a similar setting as in [3], that is, we compare the logic and the proof system with respect to their power to *distinguish non-isomorphic graphs*. We say that CPT distinguishes all graphs in a graph class $\mathcal{K}$ if there exists a polynomial resource bound $p(n)$ such that for all pairs of non-isomorphic graphs $G, H \in \mathcal{K}$, there exists a CPT-sentence with time and space bound $p(n)$ that evaluates to true in one of the graphs and false in the other one (Definition 4). A proof system distinguishes G and H if it can refute the statement "G and H are isomorphic", encoded in a natural way as a propositional formula/system of polynomial equations $P_{\text{iso}}(G, H)$ (Definition 7). Our main result reads as follows:

▶ **Theorem 1.** *Let $\mathcal{K}$ be a class of graphs such that* CPT *distinguishes all graphs in $\mathcal{K}$. Then the degree-*3 *extended polynomial calculus over $\mathbb{Q}$ (denoted* EPC_3*) distinguishes all graphs in $\mathcal{K}$ with refutations of polynomial size.*
Moreover, the EPC_3*-refutation uses only extension axioms $\frac{}{X_f - f}$ for polynomials of the form $f = X \cdot Y$ or $f = \frac{1}{n} \cdot \left(\sum_{i=1}^{n^2} X_i \right)$. That is, only monomials and certain "averaged sums" are replaced with new variables.*

This has two main consequences. Most importantly, it establishes a new potential approach for the difficult open problem of proving strong lower bounds for CPT. A central topic in finite model theory is the quest for a logic that captures Ptime (see [6, 15, 17, 25]). At the moment, CPT is arguably the most prominent candidate logic for this, after rank logic has been ruled out [22]. That is, evaluating any fixed CPT-sentence in a given input structure is in Ptime, and as of yet, no decision problem in Ptime is known that cannot be defined by a CPT-sentence. However, the isomorphism-invariance of CPT is a severe limitation. Intuitively, it means that every classical algorithm involving choices or ordered iterations, such as e.g. Gaussian elimination, has to be executed in parallel for all possible orderings of the input structure, at least if it is implemented in the naive way in CPT. This requires exponential space and time resources. Thus, CPT can only be equal to P if there exists some clever trick that allows to simulate ordered iterations in a symmetry-invariant way. One quite well-studied problem that is conjectured to be hard for CPT is solving linear equation systems over finite fields – particularly hard instances of this problem arise as encodings of the isomorphism problem of *Cai-Fürer-Immerman graphs* [5] or *multipedes* [18]. Thus, if CPT does not capture Ptime, then it is quite likely that the graph isomorphism problem on a suitable graph class is a witness for that. Unfortunately, only few techniques for proving limitations of CPT are known (essentially the symmetry-based ones employed in [10, 26, 23]). Theorem 1 opens up a new perspective, as it enables us to transfer proof-theoretic lower bounds to CPT:

▶ **Theorem 2.** *If there is a class $\mathcal{K}$ of graphs on which the isomorphism problem is in* Ptime*, but which cannot be distinguished in* EPC_3 *with polynomial-size refutations using only extension axioms of the form mentioned in Theorem 1, then* CPT $\neq$ Ptime*.*

Interesting candidate graph classes are the said CFI-graphs or multipedes; crucially, without any order relation, because on certain ordered versions of these graphs, the isomorphism problem is already known to be in CPT ([10], [1]). Recently, a super-polynomial lower bound for EPC was found [2]. It concerns the *bit-value principle* and is based on the bit-complexity of the coefficients required for a refutation. It might be a starting point in the search for

graph isomorphism lower bounds, even though it seems that its proof is not directly adaptable to this problem. A second consequence of Theorem 1, together with known results from finite model theory and proof complexity ([10, 3]), is the separation of the bounded-degree polynomial calculus and EPC_3:

▶ **Theorem 3.** *There exists a sequence of pairs of non-isomorphic graphs $(G_n, H_n)_{n\in\mathbb{N}}$ such that $P_{iso}(G_n, H_n)$ has a polynomial-size refutation in the degree-3 extended polynomial calculus (using only extension axioms of the aforementioned form) but there is no $k \in \mathbb{N}$ such that the degree-k polynomial calculus can refute $P_{iso}(G_n, H_n)$ for all n.*

To our knowledge, the separation of these two bounded-degree proof systems has not explicitly been stated before – specifically for the graph isomorphism problem. In the unbounded-degree setting, an exponential separation between PC and EPC is known, even if one only allows extension axioms of the form $\overline{X} = 1 - X$, as in *polynomial calculus resolution* [11]. The main value of Theorem 3 is that it demonstrates how finite-model-theoretic lower and upper bounds can directly lead to corresponding results in proof complexity. Other examples of finite-model-theoretic proofs for results in proof-complexity were given in [14].

2 Preliminaries

All structures in this article are finite and relational. Formally, we assume that all relations are binary (whenever we need unary relations, we encode them as binary relations). We use the words "graphs" and "binary structures" interchangeably, so in particular, graphs can be vertex- or edge-coloured. For a τ-structure A and relation symbol $R \in \tau$, $R(A)$ denotes the corresponding relation in the structure A. The universe of A is denoted $V(A)$. We need the following concepts from finite model theory:

Weisfeiler Leman algorithm. The k-dimensional Weisfeiler Leman algorithm (k-WL) is an incomplete graph isomorphism test that computes a canonical colouring of the k-tuples of vertices. Two graphs G and H are distinguished by k-WL if there is a colour class whose size is different in the colouring of G and of H. For a precise definition and a survey, see e.g. [21].

***k*-variable counting logic.** We denote by $\mathcal{C}^k$ the k-variable fragment of first-order logic augmented with counting quantifiers $\exists^{\geq i}$, for every $i \in \mathbb{N}$. For two structures G, H we write $G \equiv_{\mathcal{C}^k} H$ if G and H satisfy exactly the same $\mathcal{C}^k$-sentences. It is well-known (see Theorem 2.2 in [21]) that k-WL distinguishes G and H if and only if $G \not\equiv_{\mathcal{C}^{k+1}} H$. In fact, the colour classes of the stable k-WL colouring correspond to the $\mathcal{C}^{k+1}$-types of the k-tuples. In this paper, we are mainly concerned with 2-WL and $\mathcal{C}^3$-types of vertex-pairs in graphs. The $\mathcal{C}^3$-type of a pair (v, w) in a structure A is the collection of all $\mathcal{C}^3$-formulas $\varphi(x, y)$ such that $A \models \varphi(v, w)$. It contains a lot of (non-local) information, e.g. whether or not v and w are connected, the length of the shortest path between them, etc.

The bijective *k*-pebble game. This game is played by two players, Spoiler and Duplicator, on two structures G and H. A position of a play is a set of pebble-pairs $\pi \subseteq V(G) \times V(H)$ with $|\pi| \leq k$. In every round, Spoiler selects a subset $\pi' \subseteq \pi$ with $|\pi'| < k$ of the current pebbles, which remain on the board. Duplicator then specifies a bijection $f : V(G) \longrightarrow V(H)$. Spoiler chooses a $v \in V(G)$, leading to the new position $\pi' \cup \{(v, f(v))\}$. Spoiler wins if the pebbles do not induce a local isomorphism between the pebbled substructures. Duplicator wins if she can play infinitely avoiding the pebbling of non-isomorphic substructures. Spoiler has a winning strategy for the bijective k-pebble game on G and H if and only if $G \not\equiv_{\mathcal{C}^k} H$ [19].

Fixed-point logic with counting (FPC). FPC is a standard logic of reference in algorithmic model theory. For the purposes of this paper, it suffices to know that for every sentence $\psi \in \mathrm{FPC}$, there is a k such that whenever it holds $G \equiv_{\mathcal{C}^k} H$ for two structures, then $G \models \psi$ if and only if $H \models \psi$. See [9] for a survey on this logic and its expressive power.

3 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 [13]. The work that originally introduced CPT as an abstract state machine model is [4]; later, more "logic-like" presentations of CPT were invented, such as Polynomial Interpretation Logic (see [12, 27]) and BGS-logic [26]. In short, CPT is FPC plus a mechanism to construct isomorphism-invariant hereditarily finite sets of polynomial size. When a CPT-sentence Π is evaluated in a finite structure A, then Π may augment 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. We also write $\mathrm{CPT}(p(n))$ for the set of all CPT-sentences whose polynomial bound is at most $p(n)$. For the sake of illustration, we sketch the definition of BGS-logic:

The sentences of BGS-logic are called *programs*. A program is a tuple $\Pi = (\Pi_{\mathrm{step}}(x), \Pi_{\mathrm{halt}}(x), \Pi_{\mathrm{out}}(x), p(n))$. Here, $\Pi_{\mathrm{step}}(x)$ is a BGS-term, Π_{halt} and Π_{out} are BGS-formulas, and $p(n)$ is a polynomial that bounds the time and space used by the program. BGS-terms take as input hereditarily finite sets and output a hereditarily finite set. Examples of such terms are $\mathsf{Pair}(x, y)$, which evaluates to $\{x, y\}$, or $\mathsf{Union}(x) = \bigcup_{y \in x} y$. Furthermore, if s and t are terms, x is a variable, and φ a formula, then $\{s(x) \ : \ x \in t \ : \ \varphi(t)\}$ is a comprehension term. It applies the term s to all elements of the set defined by t that satisfy φ, and outputs the set of the resulting objects $s(x)$. When a program is evaluated in a given finite structure A, then the term $\Pi_{\mathrm{step}}(x)$ is iteratively applied to its own output, starting with $x_0 = \emptyset$. The iteration stops in step i if the computed set $x_i = (\Pi_{\mathrm{step}})^i(\emptyset)$ satisfies $A \models \Pi_{\mathrm{halt}}(x_i)$. The formula Π_{out} defines, in dependence of x_i, whether the run is accepting or rejecting, that is, whether $A \models \Pi$ or not. If the length of the run or the size of the transitive closure of x_i exceeds $p(|A|)$ at some point, then the computation is aborted, and $A \not\models \Pi$.

Recently, the computation model *Deep Weisfeiler Leman* (DWL) has been introduced by Grohe, Schweitzer and Wiebking [16]. DWL and CPT mutually simulate each other, and DWL is better suited to establish the connection to proof complexity. We present DWL in detail in Section 7, since our proof of Theorem 1 actually goes via DWL. When we say that CPT *distinguishes* certain graphs, we formally mean this:

▶ **Definition 4** (Distinguishing relational structures in CPT). *Let $\mathcal{K}$ be a class of τ-structures. We say that* CPT *distinguishes all structures in $\mathcal{K}$ if there exists a polynomial $p(n)$ and a constant $k \in \mathbb{N}$ such that for any two structures $G, H \in \mathcal{K}$ which are non-isomorphic, there exists a sentence $\Pi \in \mathrm{CPT}(p(n))$ with $\leq k$ variables such that $G \models \Pi$ and $H \not\models \Pi$.*

This definition is perhaps non-standard because we allow the distinguishing sentence to be different for every pair of graphs in $\mathcal{K}$, whereas normally, one would expect a single sentence that distinguishes all graphs in the class. Our definition, however, matches the situation in proof complexity. As we explain in the next section, a proof system distinguishes all graphs in $\mathcal{K}$, if there exists an efficient proof for non-isomorphism of each pair of non-isomorphic graphs. This proof can of course be a different one for each pair of graphs, and the distinguishing

CPT-sentences will play the role of the non-isomorphism proofs. The constant bound on the variable number is needed because with an unbounded number of variables, we could already find a first-order sentence for every graph that describes it up to isomorphism. Later on, in the DWL framework, this variable bound becomes irrelevant because DWL algorithms naturally correspond to bounded-variable CPT programs.

The distinguishing power of CPT is strictly greater than that of FPC. This can for example be seen by considering CFI-graphs over linearly ordered *base graphs* (every CFI-graph is obtained by applying the construction from [5] to a given connected base graph). We can summarise the situation like this:

▶ **Theorem 5** ([10]). *There is a family of pairs of graphs $(G_n, H_n)_{n\in\mathbb{N}}$, that are equipped with a total preorder on the vertex set (encoded as a binary relation $\preceq$), such that:*

- *$G_n \not\cong H_n$ for all $n \in \mathbb{N}$.*
- *For every* FPC*-sentence ψ and all large enough $n \in \mathbb{N}$: $G_n \models \psi$ if and only if $H_n \models \psi$.*
- *There is a* CPT*-sentence Π such that for all $n \in \mathbb{N}$: $G_n \models \Pi$ and $H_n \not\models \Pi$.*

4 The (extended) polynomial calculus

The *polynomial calculus* (PC) was introduced in [8]. It is applicable to the following problem: Given a set P of multivariate polynomials over a fixed field (in our case, $\mathbb{Q}$), decide if the polynomials in P have a common zero with respect to $\{0,1\}$-assignments. The polynomial 1 is derivable from P if and only if the polynomials in P have no common zero over $\{0,1\}$. A derivation of the 1-polynomial is formally a sequence $p_1, p_2, ..., p_n = 1$ of polynomials such that each p_i is either in P or an axiom of the polynomial calculus or is obtained from one or multiple p_j, for $j < i$, with the application of one of the derivation rules listed below. A derivation of the 1-polynomial from P is called a *refutation* of P.
A restricted variant of the polynomial calculus, the monomial calculus, has been introduced in [3]. The derivation rules of the polynomial/monomial calculus are the following:

▶ **Definition 6** (Inference rules of the (extended) polynomial calculus). *Let P be the set of input polynomials/axioms, $a, b \in \mathbb{Q}$, X a variable, and f, g polynomials with rational coefficients.*

$$\frac{p \in P}{p}\ \textit{(Axioms)} \qquad \frac{}{X^2 - X}\ \textit{(Boolean axioms)}$$

$$\frac{f}{Xf}\ \textit{(Multiplication rule)} \qquad \frac{g \quad f}{ag + bf}\ \textit{(Linear combination rule)}$$

In the extended polynomial calculus *(*EPC*),* extension axioms *of the form $\frac{}{X_f - f}$ may be used whenever X_f is a fresh variable not occurring in f. The Boolean axioms do not apply to these extension variables.*

The *monomial calculus* (MC) is a restriction of PC that permits the use of the multiplication rule only in the cases where f is either a monomial or the product of a monomial and an axiom. For MC, PC, and EPC, we also consider the degree-k restrictions denoted MC_k, PC_k, and EPC_k. Proofs in these degree-restricted calculi may only consist of polynomials of degree at most k. In general, these proof systems are not complete any more, but bounding the degree by a constant yields natural fragments that admit efficient proof search via Gröbner basis computation (at least for MC_k and PC_k, this is the case; see [8]).
The *size* of a refutation $p_1, p_2, ..., p_n = 1$ is the total number of occurrences of monomials in all its polynomials. Its *bit-complexity* is the maximum number of bits required to represent any of the occurring coefficients, where values in $\mathbb{Q}$ are stored as a fraction of two binary numbers.

Intuitively, the effect of the extension axioms in the bounded-degree setting is that the degree-bound of three may be "locally" violated: Monomials like $A \cdot B \cdot C \cdot D$ can be written as $X_{AB} \cdot X_{CD}$, where X_{AB} and X_{CD} are fresh extension variables such that $X_{AB} = A \cdot B$, and $X_{CD} = C \cdot D$. Thus, with the help of extension axioms, we can implicitly use monomials of larger degree than allowed. If we restrict ourselves to refutations of polynomial size, then we can think of EPC_3 as a version of degree-3 polynomial calculus where the degree bound can be violated a limited number of times.

5 Applying algebraic proof systems to the graph isomorphism problem

Let G and H be fixed graphs, potentially with a colouring of the vertices or with multiple edge relations. We consider the following polynomial axiom system $P_{\mathrm{iso}}(G, H)$ that expresses the existence of a (colour-preserving) isomorphism between G and H. A refutation of $P_{\mathrm{iso}}(G, H)$ in any variant of the polynomial calculus then witnesses that G and H are non-isomorphic. This definition of $P_{\mathrm{iso}}(G, H)$ is almost the same as in [3].

▶ **Definition 7** ($P_{iso}(G, H)$, [3]). *Let G and H be two graphs (potentially vertex-coloured). Let $\sim \subseteq V(G) \times V(H)$ be the relation "vertex $v \in V(G)$ and $w \in V(H)$ have the same colour". The system $P_{iso}(G, H)$ consists of the following polynomials in the variables $\{X_{vw} \mid v \in V(G), w \in V(H), v \sim w\}$.*

$$\sum_{\substack{v \in V(G) \\ v \sim w}} X_{vw} - 1 \qquad \text{for all } w \in V(H) \tag{1}$$

$$\sum_{\substack{w \in V(H) \\ v \sim w}} X_{vw} - 1 \qquad \text{for all } v \in V(G) \tag{2}$$

$$X_{vw} X_{v'w'} \qquad \begin{array}{l} \text{for all } v, v' \in V(G), w, w' \in V(H) \\ \text{with } v \sim w \text{ and } v' \sim w' \\ \text{such that } \{(v, w), (v', w')\} \text{ is not} \\ \text{a local isomorphism.} \end{array} \tag{3}$$

The intended meaning of the variable X_{vw} being set to one is "v is mapped to w". When we say that a certain variant of the polynomial calculus *distinguishes* two graphs G, H, we mean that the polynomial equation system $P_{\mathrm{iso}}(G, H)$ has a refutation in that proof system. The main result from [3] links graph distinguishability in this sense to graph distinguishability by the k-dimensional Weisfeiler Leman algorithm.

▶ **Theorem 8** (Theorem 4.4 in [3]). *Let $k \in \mathbb{N}$ and let G and H be graphs. The axiom system $P_{iso}(G, H)$ has a refutation in the degree-k monomial calculus iff the $(k-1)$-dimensional Weisfeiler Leman algorithm distinguishes G and H.*

In [3], this is not stated for vertex- or edge-coloured graphs, but it can be checked that the proof still goes through in these cases. For our result, we need some of the technical ingredients from the proof of Theorem 8: What is shown in [3] is that Spoiler's winning positions in the bijective k-pebble game on G and H are derivable in MC_k from $P_{\mathrm{iso}}(G, H)$. A position in the game is a set of pebble pairs $\pi \subseteq V(G) \times V(H)$ of size $|\pi| \leq k$. The position π corresponds to a monomial in the variables from $P_{\mathrm{iso}}(G, H)$. We denote this monomial as $X_\pi := \prod_{(v,w) \in \pi} X_{vw}$ (so the X_{vw} are the variables, whereas X_π is shorthand for a *product* of variables). We will use the following central technical result as a blackbox:

▶ **Lemma 9** (Lemma 4.2 in [3]). *Let $k \geq 2$ and G, H be graphs (such that for every vertex-colour Q, there are exactly as many vertices of colour Q in G as in H). If Spoiler has a winning strategy for the bijective k-pebble game on G, H with initial position π, then there is an MC_k-derivation of the monomial X_π from $P_{iso}(G, H)$.*

This lemma accounts for one direction of Theorem 8 because if $G \not\equiv_{\mathcal{C}^k} H$, then Spoiler wins the bijective k-pebble game from the initial position $\emptyset$, and we have $X_\emptyset = 1$.

6 Separating the bounded-degree extended polynomial calculus from its non-extended version

Before we come to the more technical part, we show how Theorem 3 follows from Theorem 1. According to Theorem 6.2 in [3], for every $n \in \mathbb{N}$, there exist pairs $G_n, \widetilde{G}_n$ of non-isomorphic CFI-graphs of size $\mathcal{O}(n)$ such that $P_{\text{iso}}(G_n, \widetilde{G}_n)$ has no degree-n polynomial calculus refutation (over $\mathbb{Q}$). A closer examination of the construction in [3] reveals that the axioms (1) and (2) in $P_{\text{iso}}(G_n, \widetilde{G}_n)$ are for coloured versions of the respective CFI-graphs: Each vertex-gadget and each edge-gadget of G_n and $\widetilde{G}_n$, respectively, forms a distinct colour class and the axioms restrict possible isomorphisms to colour-preserving ones (the precise definition of the said vertex- and edge-gadgets is not essential here, so we refer to [10] for the presentation of the CFI construction). Therefore, we have $P_{\text{iso}}(G_n, \widetilde{G}_n) = P_{\text{iso}}((G_n, \preceq), (\widetilde{G}_n, \preceq))$ for any preorder $\preceq$ on $V(G_n)$ that is obtained from a linear order on the respective base graph; in other words, a preorder that linearly orders the CFI-gadgets without ordering the vertices inside each gadget (for details, see [5] or [10]). Note that our definition of P_{iso} also works for graphs with multiple edge relations, and we can simply view the binary relation $\preceq$ as another type of edge relation. Now the system $P_{\text{iso}}((G_n, \preceq), (\widetilde{G}_n, \preceq))$ does have a polynomial-size refutation in EPC_3, for any choice of the ordering, because CFI-graphs over linearly ordered base graphs can be distinguished in CPT [10] and thus, a refutation exists by our Theorem 1.

7 Deep Weisfeiler Leman

Before we are ready to prove Theorem 1, we have to introduce the technical details of *Deep Weisfeiler Leman*, a computation model equivalent to CPT that we will simulate in EPC_3. A DWL-algorithm is a deterministic Turing machine that gets as input a finite structure with binary relations. All DWL-computations are isomorphism-invariant: The machine does not have access to the input structure directly, but only to its so-called *algebraic sketch*. This is a certain invariant of the structure, similar to its 2-dimensional Weisfeiler Leman colouring. The machine is not only able to read information about its input structure but it can also modify the structure in an isomorphism-invariant way. These modifications correspond to the creation of higher-order objects in Choiceless Polynomial Time.

Before we can introduce the Deep Weisfeiler Leman framework in detail, we have to say precisely what the algebraic sketch of a structure is. It is a representation of its *coarsest coherent configuration* (or "coherent colouring"), that is defined below. The coarsest coherent configuration of a structure is also known as its stable 2-dimensional Weisfeiler Leman colouring (equivalent to the partition of all pairs into their $\mathcal{C}^3$-types). The following presentation closely follows the one in [16].

7.1 Coherent configurations of binary structures

▶ **Definition 10** (Notions concerning binary relations, [16]).

- *The* converse *of a relation R is the relation $R^{-1} := \{(v,u) \mid (u,v) \in R\}$.*
- *For a set V, the* diagonal *of V is the relation* $\mathrm{diag}(V) := \{(v,v) \mid v \in V\}$. *For a relation $R \subseteq V^2$ we let $R^{\mathrm{diag}} := R \cap \mathrm{diag}(V)$ be the diagonal elements in R. We call R a* diagonal relation *if $R = R^{\mathrm{diag}}$.*
- *The* strongly connected components *(*SCC*s) of a relation R are defined in the usual way as inclusionwise maximal sets S such that for all $u, v \in S$ there is an R-path of length at least 1 from u to v. (A singleton set $\{u\}$ can be a strongly connected component only if $(u,u) \in R$.) We write $\mathrm{scc}(R)$ to denote the set of strongly connected components of R. Moreover, we let $R^{\mathrm{scc}} := \bigcup_{S \in \mathrm{scc}(R)} S^2$ be the relation describing whether two elements are in the same strongly connected component.*

▶ **Definition 11** (Coherent configurations, [16]). *Let σ be a vocabulary. A* coherent σ-configuration *C is a σ-structure C with the following properties.*

- *$\{R(C) \mid R \in \sigma\}$ is a partition of $V(C)^2$.*
- *For each $R \in \sigma$ the relation $R(C)$ is either a subset of or disjoint from the diagonal $\mathrm{diag}(V(C))$.*
- *For each $R \in \sigma$ there is an $R^{-1} \in \sigma$ such that $R^{-1}(C) = (R(C))^{-1}$.*
- *For all $R_1, R_2, R_3 \in \sigma$ there is a number $q = q(R_1,R_2,R_3) \in \mathbb{N}$ such that for all $(u,v) \in R_1(C)$ there are exactly q elements $w \in V(C)$ such that $(u,w) \in R_2(C)$ and $(w,v) \in R_3(C)$.*

The numbers $q(R_1,R_2,R_3)$ are called the intersection numbers *of C and the function $q : \sigma^3 \to \mathbb{N}$ is called the* intersection function.

A coherent σ-configuration C is at least as *fine* as, or *refines*, a τ-structure A (we write $C \sqsubseteq A$) if $V(A) = V(C)$, and for each $R \in \sigma$ and each $E \in \tau$ it holds that $R(C) \subseteq E(A)$ or that $R(C) \subseteq A^2 \setminus E(A)$. We say that a coherent configuration C is a *coarsest coherent configuration refining* a structure A if $C \sqsubseteq A$ and $C' \sqsubseteq C$ for every coherent configuration C' satisfying $C' \sqsubseteq A$. In the following, we will usually write τ for the vocabulary of a given structure and σ for the vocabulary of the corresponding coherent configuration, without further specifying σ.

Every binary structure A has a coarsest coherent configuration refining it, which can be computed efficiently with the 2-dimensional Weisfeiler Leman algorithm (Theorem 2.1 in [16]). This configuration is unique up to the renaming of relation symbols. We write $C(A)$ for the coarsest coherent configuration of A with canonical names of the relation symbols, as for example produced by a fixed implementation of 2-WL. We call the relation symbols in σ *colours* to distinguish them from the relation symbols in τ. In the following, we often identify the symbols in τ and σ with binary strings, because this is how they are represented in a Turing machine.

The *algebraic sketch* of a structure contains information about the colours appearing in its coarsest coherent configuration, which relations of the structure they refine, and the intersection function q. Formally, the algebraic sketch of a structure A is the tuple $D(A) = (\tau, \sigma, \subseteq_{\sigma,\tau}, q)$. The relation $\subseteq_{\sigma,\tau}$ relates the colours in σ with the relations in τ they refine: $\subseteq_{\sigma,\tau} := \{(R,E) \in \sigma \times \tau \mid R(C(A)) \subseteq E(A)\}$. In order to feed $D(A)$ to a Turing machine, we have to agree on some encoding in binary. If the binary string encodings of the relation symbols are fixed, then there is a canonical encoding of $D(A)$, based on the lexicographic ordering of the relation names and ordering of the intersection numbers $q(R_1,R_2,R_3)$. The string that encodes $D(A)$ is the initial tape content in a DWL-computation on the structure A.

7.2 The Deep Weisfeiler Leman computation model

A DWL-algorithm is a two-tape Turing machine M with an additional storage device that the authors of [16] have named "the *cloud*". It contains a structure A together with its coarsest coherent configuration $C(A)$. The storage that the machine itself can use is a *work tape* and an *interaction tape*, which allows for interaction with the cloud. The input of a DWL-program M is a binary τ-structure A. Initially, the cloud contains the coherently coloured structure $(A, C(A))$, and on the interaction tape, the algebraic sketch $D(A)$ is written. The work tape is empty.

The Turing machine works as a standard Turing machine with two special transitions that can modify the structure in the cloud. To execute these, the machine writes a binary string $s \in \{0,1\}^*$ on the interaction tape and enters one of the two distinguished states $q_{\texttt{addPair}}, q_{\texttt{contract}}$. If s is a colour $R \in \sigma$, then we say that M executes $\texttt{addPair}(R)$ or $\texttt{contract}(R)$. Executing $\texttt{addPair}(R)$ creates a new vertex for each vertex-pair whose colour in $C(A)$ is R. The operation $\texttt{contract}(R)$ contracts every SCC formed by pairs of colour R into a single vertex.

- $\texttt{addPair}(R)$: The machine adds a fresh vertex for each pair in $R(C(A))$ to A, i.e. it updates $V(A)$ to $V(A) \uplus R(C(A))$. These pairs are then connected with their elements in A. Therefore, τ is updated to $\tau \cup \{E_{\text{left}}, E_{\text{right}}\} \uplus \{D_R\}$. The new relation D_R identifies the newly added vertices, i.e. $D_R(A) := \text{diag}(R(C(A)))$. The symbol D_R is chosen as the lexicographically smallest binary string that is not yet used as a relation symbol.
  Furthermore, $E_{\text{left}}(A)$ is updated to $E_{\text{left}}(A) \cup \{(u,(u,v)) \in V(A)^2 \mid (u,v) \in R(C(A))\}$, and $E_{\text{right}}(A)$ is set to $E_{\text{right}}(A) \cup \{(v,(u,v)) \in V(A)^2 \mid (u,v) \in R(C(A))\}$ (where initially, $E_{\text{left}}(A) = E_{\text{right}}(A) = \emptyset$).
- $\texttt{contract}(R)$: Let $\mathcal{S} := \text{scc}(R)$ be the set of strongly connected components of the relation R. Let $U := V(A) \setminus \bigcup \mathcal{S}$ be the set of vertices that are not in one of the strongly connected components. The components in $\mathcal{S}$ are contracted. That means we update $V(A)$ to $U \uplus \mathcal{S}$, and τ to $\tau \uplus \{D_R\}$. Again, $D_R(A) := \text{diag}(\mathcal{S})$. For each relation $E \in \tau$, we update $E(A)$ to $(E(A) \cap U^2) \cup \{(u,S) \mid \exists v \in S \in \mathcal{S} : (u,v) \in E(A)\} \cup \{(S,v) \mid \exists u \in S \in \mathcal{S} : (u,v) \in E(A)\} \cup \{(S_1,S_2) \mid \exists u \in S_1 \in \mathcal{S}, \exists v \in S_2 \in \mathcal{S} : (u,v) \in E(A)\}$.

Each of these special transitions modifies the structure A in the cloud in an isomorphism-invariant way. After that, the cloud storage device computes $C(A)$ (for example, with the 2-WL algorithm) and stores the coherently coloured structure $(A, C(A))$. The algebraic sketch $D(A)$ of the new structure is written on the interaction tape.

A DWL-algorithm decides a class $\mathcal{K}$ of τ-structures in the usual sense, i.e. the algorithm halts with output 1 on input A if $A \in \mathcal{K}$, and else, it halts with output 0. We say that a DWL-algorithm *runs in polynomial time* if the number of computation steps of the Turing machine and the size of the structure in the cloud is bounded by a polynomial in the size of the input structure. The original definition of DWL in [16] also has two more operations, `create` and `forget`, but it can be shown that these do not increase the expressive power; the proof is similar to the proof in [16] showing that "pure DWL" simulates DWL, so we omit it.

7.3 Distinguishing graphs in Deep Weisfeiler Leman

In [16], the authors say that a DWL-algorithm *decides isomorphism* on a structure class $\mathcal{K}$ if it gets as input the disjoint union $A := G \uplus H$ of two connected binary structures and correctly decides whether $G \cong H$. A crucial technical result in [16] shows that one can always assume that at any stage of the computation, the structure in the cloud is the disjoint union of two connected structures: It is never necessary for the algorithm to produce connections

between the two components, i.e. `addPair` and `contract` are only executed for colours R with $R(C(A)) \subseteq V(G)^2 \cup V(H)^2$. A DWL-algorithm that maintains this invariant is called *normalised* in [16].

▶ **Definition 12** (Distinguishing structures in DWL). *The computation model* DWL distinguishes *all structures in a class $\mathcal{K}$ (of connected τ-structures) in* polynomial time *if: There is a polynomial $p(n)$ such that for any two non-isomorphic structures $G, H \in \mathcal{K}$, there exists a* normalised *DWL-algorithm M which, given $G \uplus H$ as input, terminates with a structure $G' \uplus H'$ in the cloud such that $D(G') \neq D(H')$, and takes time and space at most $p(|G|+|H|)$.*

This is simply the DWL-version of Definition 4 for distinguishing structures in CPT. The main difference to the CPT-setting is that here, the constant bound on the number of variables is already implicit in the definition of DWL (because DWL only accesses a structure via its coherent configuration, and two structures with the same configuration are $\mathcal{C}^3$-equivalent). Let us elaborate on what is meant precisely by $D(G') \neq D(H')$. Whenever $A = A_1 \uplus A_2$ is a τ-structure consisting of two separate connected components, then we write $D(A)[A_1]$ and $D(A)[A_2]$ for the restrictions of the algebraic sketch $D(A)$ to the σ-colours that occur as colours of pairs in $V(A_1)^2$ or $V(A_2)^2$, respectively. It follows from the properties of normalised DWL-computations (Lemma 8 in [16]) that $D(A)[A_i]$ is in fact the algebraic sketch of A_i, so the sketch of $A_1 \uplus A_2$ is composed of the sketches of the two structures:

▶ **Lemma 13.** *Let $A = A_1 \uplus A_2$ be the disjoint union of two connected τ-structures. For $i \in \{1, 2\}$, $D(A)[A_i]$ is an algebraic sketch and equal to $D(A_i)$, up to a renaming of the colours in σ.*

Thus, when we write $D(G') \neq D(H')$, we are formally referring to the respective restrictions of $D(G' \uplus H')$, but these are equivalent to $D(G')$ and $D(H')$, respectively. The algebraic sketches being distinct means that 2-WL distinguishes the structures. By standard results (see e.g. [21]), one can infer:

▶ **Lemma 14.** *Let G, H be connected τ-structures. It holds $D(G) \neq D(H)$ if and only if Spoiler has a winning strategy for the bijective 3-pebble game on G and H.*

We can say even more, namely that Spoiler can distinguish pairs of different colours. The following is a variation of a standard result (Theorem 2.2 in [21]). The standard result concerns the setting where the graphs G and H are considered separately with their respective coarsest coherent configurations. Here, we have to work with their disjoint union. It is perhaps not surprising that in this setting, the correspondence between Weisfeiler-Leman colourings and pebble games also exists, but we are not aware of a formal proof for this statement for $G \uplus H$ in the literature. Thus, for completeness, we provide one in the full version of the paper [24].

▶ **Lemma 15.** *Let G, H be two connected τ-structures and $A := G \uplus H$ with its coarsest coherent σ-configuration $C(A)$. Let $(v, v') \in V(G)^2, (w, w') \in V(H)^2$ such that there is no $R \in \sigma$ with $(v, v') \in R(C(A))$ and $(w, w') \in R(C(A))$.*
Then Spoiler has a winning strategy for the bijective 3-pebble game on G and H with initial position $\{(v, w), (v', w')\}$.

As shown below, if CPT distinguishes all structures in a class $\mathcal{K}$, then also DWL distinguishes all structures in $\mathcal{K}$ in polynomial time. Hence, in our proof of Theorem 1, we can indeed start with the assumption that DWL polynomially distinguishes all graphs in $\mathcal{K}$.

▶ **Lemma 16.** *Let $\mathcal{K}$ be a class of connected structures that are distinguished by* CPT *in the sense of Definition 4. Then DWL distinguishes all structures in $\mathcal{K}$ in polynomial time in the sense of Definition 12.*

Proof sketch. Let $p(n)$ be the resource bound for the distinguishing CPT-programs for the class $\mathcal{K}$ that exists by Definition 4. Fix two τ-structures $G, H \in \mathcal{K}$ such that $G \ncong H$. Let $\Pi \in \mathrm{CPT}(p(n))$ be a distinguishing sentence. By Theorem 21 in [16], there exists a polynomial time DWL-algorithm M which simulates Π (and the polynomial resource bound of M depends only on $p(n)$, not on G and H). That means M w.l.o.g. accepts G and rejects H. The sequence of executed cloud-interaction operations from $\{\texttt{addPair}, \texttt{contract}\}$ is the same in the run of M on G as in the run on H, up to the point where the respective structures in the cloud have distinct algebraic sketches (because the behaviour of M only depends on the algebraic sketch of the structure in the cloud). At that point, we can stop the simulation of Π by M because we do not actually care about the acceptance behaviour as long as the machine produces distinct sketches on G and H. Now the same sequence of cloud-interaction-operations can be simulated by an appropriate DWL-algorithm M' on input $G \uplus H$, which leads to a structure $G' \uplus H'$ with $D(G') \neq D(H')$. Such an M' can be constructed because of Lemma 13. ◀

8 Properties of coherent configurations

Here is a small collection of lemmas concerning coherent configurations. We will need them in our construction of the EPC_3-refutation in the next section.

▶ **Lemma 17.** *Let A be a τ-structure and $C(A)$ its coarsest coherent σ-colouring. Let $R \in \sigma$. There are diagonal colours $D_1, D_2 \in \sigma$ such that for all pairs $(u, v) \in R(C(A))$ we have $(u, u) \in D_1(C(A))$, and $(v, v) \in D_2(C(A))$.*

Proof. This is Corollary 2.1.7 in [7] (the term "fibers" there means the same as our "diagonal colours"). ◀

▶ **Corollary 18.** *Let A be a τ-structure and $C(A)$ its coarsest coherent σ-colouring. Let $R \in \sigma$. If for any $(u, v) \in R(C(A))$, (u, u) or (v, v) is in some relation $E(A)$, for $E \in \tau$, then for all other $(u', v') \in R(C(A))$, it also holds $(u', u') \in E(A)$, or $(v', v') \in E(A)$, respectively.*

Proof. Follows from the previous lemma and the fact that the coarsest coherent configuration of A is a refinement of the relations of A. ◀

▶ **Lemma 19.** *Let A be a τ-structure and let $C(A)$ be its coarsest coherent σ-configuration. Let $R \in \sigma$ and let $\mathcal{S} = \mathrm{SCC}(R)$ be the set of R-SCCs in $C(A)$. There is a diagonal relation $P \in \sigma$ such that* $\mathrm{diag}(\bigcup \mathcal{S}) = P(C(A))$.

Proof. First, we show that $\mathrm{diag}(\bigcup \mathcal{S}) \subseteq P(C(A))$. Any vertex $v \in \bigcup \mathcal{S}$ has some outgoing R-neighbour w that is in the same SCC (possibly, $w = v$). That is, we have $(v, w) \in R(C(A))$. By Lemma 17, there are specific diagonal relations D_1, D_2 such that $(v, v) \in D_1(C(A)), (w, w) \in D_2(C(A))$, and all endpoints of R-edges have these diagonal colours. But since w is itself the left entry in some other R-edge (w, w'), we must have $D_1 = D_2 =: P$.
It remains to prove $P(C(A)) \subseteq \mathrm{diag}(\bigcup \mathcal{S})$. For any $v \in \mathrm{diag}(\bigcup \mathcal{S})$, there exists an R-path of length ≥ 1 from v to itself. We have already argued that $(v, v) \in P(C(A))$. It follows that any other vertex w with $(w, w) \in P(C(A))$ also has an R-path to itself and is thus in $\bigcup \mathcal{S}$. To see this, recall that $C(A)$ corresponds to the stable 2-WL-colouring [16], which in turn

partitions A^2 into $\mathcal{C}^3$-types [21]. Hence, all vertices with the same diagonal colour satisfy exactly the same $\mathcal{C}^3$-formulas. The existence of an R-path from a vertex to itself (in the fixed structure A) is expressible in $\mathcal{C}^3$ using standard techniques: Namely, for every fixed number $d \leq |A|$, we can write a $\mathcal{C}^3$-formula $\varphi_d(x, y)$ that asserts the existence of a path of length d from x to y. Only 3 variables are needed because one can alternately requantify used variables (see e.g. Proposition 3.2 in [20]). In the formula, we have access to the relation R because it is itself $\mathcal{C}^3$-definable: Essentially, R is a $\mathcal{C}^3$-type of vertex-pairs in A, and it is known that on finite structures, such a type is definable with a single formula. ◀

▶ **Corollary 20.** *Let A be a τ-structure such that for every $v \in V(A)$, (v, v) is in exactly one diagonal relation $P(A)$, for $P \in \tau$ (i.e. A is a graph with vertex colours). Let $C(A)$ be the coarsest coherent σ-configuration of A. Let $R \in \sigma$ and let $\mathcal{S} = \mathrm{SCC}(R)$. There is a colour (i.e. a diagonal relation) $P \in \tau$ such that for every SCC $S \in \mathcal{S}$,* $\mathrm{diag}(S) \subseteq P(A)$.

Proof. The coarsest coherent configuration $C(A)$ is a refinement of A. Therefore, the diagonal relations in $C(A)$ are subsets of the diagonal relations in A. Now the statement follows directly from Lemma 19. ◀

▶ **Lemma 21.** *Let $A, C(A), R \in \sigma$, and $\mathcal{S}$ be as above. All R-SCCs in $\mathcal{S}$ are of equal size.*

Proof. For any number $k \in \mathbb{N}$, we can write a $\mathcal{C}^3$-formula $\varphi_k(x)$ asserting that the size of the R-SCC of x is exactly k. To do this, we can just use a counting quantifier and the fact that the existence of an R-path between two vertices (and back) is $\mathcal{C}^3$-definable (see proof of Lemma 19). Now since all vertices in R-SCCs have the same diagonal colour (Lemma 19), and colours coincide with $\mathcal{C}^3$-types, they all satisfy the same φ_k and thus, all SCCs have equal size. ◀

9 Refuting graph isomorphism in the extended polynomial calculus - Proof of Theorem 1

Let $\mathcal{K}$ be a class of connected binary structures such that CPT distinguishes all structures in $\mathcal{K}$. By Lemma 16, then also DWL distinguishes all structures in $\mathcal{K}$ in polynomial time. Now Theorem 1 follows from Lemma 23 below that establishes the link between DWL-distinguishability and the extended polynomial calculus. Before we can prove Lemma 23, we have to state the key technical result that it depends on:

▶ **Lemma 22.** *Let G, H be two connected binary τ-structures, which are potentially vertex-coloured in such a way that for every vertex-colour Q, there are as many vertices with colour Q in G as in H. Let* $\mathrm{op} \in \{\texttt{addPair}, \texttt{contract}\}$ *and let $R \in \sigma$, where σ is the vocabulary of the coarsest coherent configuration $C := C(G \uplus H)$. Assume that $R(C) \subseteq V(G)^2 \cup V(H)^2$. Let $G' \uplus H'$ be the result of executing* $\mathrm{op}(R)$ *on $G \uplus H$.*

Then the polynomial axiom system $P_{iso}(G', H')$ is derivable from $P_{iso}(G, H)$ in EPC_3*, up to a renaming of variables. The number of extension variables used in the derivation is at most $|V(G')|^2$, and the derivation has polynomial size and uses only coefficients with polynomial bit-complexity. Moreover, for every extension axiom $\frac{}{X_f - f}$ used in the derivation, f is of the form $f = X \cdot Y$ or $f = \frac{1}{n} \cdot \left(\sum_{i=1}^{n^2} X_i\right)$.*

The proof is quite lengthy and would interrupt the proof of Theorem 1 at this point; therefore, we first present the lemma and proof that explains how Theorem 1 follows from Lemma 22. Afterwards, we provide the actual polynomial calculus derivations whose existence is claimed in Lemma 22.

▶ **Lemma 23.** *Let G, H be two connected binary τ-structures. Let $p(n)$ be a polynomial and M be a normalised* DWL*-algorithm which produces on input $G \uplus H$ a structure $G' \uplus H'$ with $D(G') \neq D(H')$, such that the length of the run and the size of the structure in the cloud is bounded by $p(|G| + |H|)$ at any time.*
Then the system $P_{iso}(G, H)$ has an EPC_3*-refutation that uses at most $p(|G| + |H|)^3$ many extension variables, has polynomial size and polynomial bit-complexity. Moreover, for every extension axiom $\frac{}{X_f - f}$ used in the derivation, f is of the form $f = X \cdot Y$ or $f = \frac{1}{n} \cdot \left(\sum_{i=1}^{n^2} X_i \right)$.*

Proof. Follows from Lemma 22 together with Lemma 14 and Lemma 9. In detail: Let $(\mathrm{op}_i(R_i))_{i \leq t}$ with $\mathrm{op}_i \in \{\texttt{addPair}, \texttt{contract}\}$ be the sequence of cloud-interaction-operations in the run of M on $G \uplus H$. This sequence of operations produces a sequence of structures $(G \uplus H, G_1 \uplus H_1, G_2 \uplus H_2, ..., G_t \uplus H_t)$ such that $D(G_t) \neq D(H_t)$.
For each i, R_i is a colour in the coarsest coherent configuration of the current structure $A_i := G_i \uplus H_i$ in the cloud. Since M is normalised, $R(C(A_i)) \subseteq V(G_i)^2 \cup V(H_i)^2$. Thus, we can inductively apply Lemma 22 to derive in EPC_3 polynomial axiom systems $P_{\mathrm{iso}}(G_i, H_i)$ for every $i \in [t]$. The induction requires that for every vertex-colour (i.e. diagonal relation), the colour classes always have the same size in G_i and H_i (this is a prerequisite of Lemma 22). This is satisfied because $D(G_i) = D(H_i)$, for $i < t$. Since $D(G_t) \neq D(H_t)$, it follows from Lemma 14 and Lemma 9 that the 1-polynomial is derivable from $P_{\mathrm{iso}}(G_t, H_t)$ in the degree-3 monomial calculus, so in total, it is derivable from $P_{\mathrm{iso}}(G, H)$ in EPC_3. In order to apply Lemma 9 to G_t and H_t, we need to argue that the vertex-colour-classes are of equal size in both graphs, even though $D(G_t) \neq D(H_t)$. If step t is $\texttt{addPair}(R)$, then we introduce equally many new pair-vertices in both graphs because $D(G_{t-1}) = D(H_{t-1})$, so the numbers of R-pairs are equal. If step t is $\texttt{contract}(R)$, we also produce the same number of new vertices. Namely, the number of R-SCCs in $C(G_{t-1})$ and $C(H_{t-1})$ is equal, because by Lemma 21, all R-SCCs have equal size, and by Lemma 19, vertices in R-SCCs receive the same diagonal colour P distinct from all diagonal colours outside SCCs (and $|P(C(G_{t-1}))| = |P(C(H_{t-1}))|$).

Finally, we bound the number of extension variables used in the derivation of $P_{\mathrm{iso}}(G_t, H_t)$: As stated in Lemma 22, for every i, the derivation of $P_{\mathrm{iso}}(G_i, H_i)$ from $P_{\mathrm{iso}}(G_{i-1}, H_{i-1})$ uses at most $|V(G_i)|^2$ many new extension variables. Therefore, the total number of extension variables that are used in the derivation of $P_{\mathrm{iso}}(G_t, H_t)$ is at most $\sum_{i \in [t]} |V(G_i)|^2$. Since $t \leq p(|G|+|H|)$ and $|V(G_i)| \leq p(|G|+|H|)$, for every $i \in [t]$, this sum is at most $p(|G|+|H|)^3$. Similarly we can bound the size and bit-complexity of the derivation: Each time we invoke Lemma 22, we only incur a polynomial cost in size, and this happens polynomially many times. The occurring coefficients can be encoded with polynomially many bits as the lemma asserts. Also, Lemma 22 uses only extension axioms of the required form. ◀

Proof of Lemma 22. First, we have to explain how the variables of the new system $P_{\mathrm{iso}}(G', H')$ are encoded as polynomials in the old variables. Recall that the variable set of $P_{\mathrm{iso}}(G, H)$ is

$$\mathcal{V}(G, H) := \{X_{vw} \mid v \in V(G), w \in V(H), v \text{ and } w \text{ have the same vertex-colour}\}.$$

The intended meaning of X_{vw} is "v is mapped to w". The graphs G', H' contain new vertices, which either represent contracted R-SCCs or pairs of colour R. The set of vertex-pairs (v, w) for which we need new variables is $\mathbf{NewPairs} := (V(G') \setminus V(G)) \times (V(H') \setminus V(H))$.
We would like to map each $(v, w) \in \mathbf{NewPairs}$ to a polynomial $f(vw)$ such that we can represent variables X_{vw} for $(v, w) \in \mathbf{NewPairs}$ as extension variables $X_{f(vw)}$, which we can introduce with the extension axiom $X_{f(vw)} = f(vw)$. If $\mathrm{op} = \texttt{addPair}$ and v is a new pair-vertex, then we let $\mathrm{pair}(v)$ be the vertex-pair that v corresponds to. If $\mathrm{op} = \texttt{contract}$ and v is a new SCC-vertex, then we let $\mathrm{scc}(v)$ denote the set of vertices in the SCC that is contracted into v. We define f as the following injective mapping $f : \mathbf{NewPairs} \to \mathbb{Q}[\mathcal{V}(G, H)]$.

$$f(vw) := \begin{cases} X_{v_1w_1}X_{v_2w_2} & \text{, if op} = \texttt{addPair} \text{ and} \\ & (v_1, v_2) = \text{pair}(v), (w_1, w_2) = \text{pair}(w). \\ \frac{1}{|\text{scc}(v)|} \cdot \sum_{(v',w') \in \text{scc}(v) \times \text{scc}(w)} X_{v'w'} & \text{, if op} = \texttt{contract}. \end{cases}$$

Note that $f(vw)$ is indeed always a polynomial in variables $\mathcal{V}(G, H)$: To see this, we have to check that in the pair-case, the vertex-colours of v_1, w_1 and of v_2, w_2, respectively, are equal, and in the SCC-case, the vertex-colours of all elements of $\text{scc}(v)$ and $\text{scc}(w)$ are equal. In the pair-case, this follows from Corollary 18, and in the SCC-case from Corollary 20.

If v and w are newly introduced pair-vertices with $\text{pair}(v) = (v_1, v_2)$ and $\text{pair}(w) = (w_1, w_2)$, then the variable X_{vw} will be the extension variable for the monomial $X_{v_1w_1}X_{v_2w_2}$. This makes sense because if X_{vw} is set to 1, then the bijection encoded by the assignment maps v to w; but then it also has to map v_1 to w_1 and v_2 to w_2. Similarly, if v and w are new SCC-vertices, then any bijection that takes v to w must also map the elements of $\text{scc}(v)$ to the elements of $\text{scc}(w)$ in any possible way. This is reflected in our representation of X_{vw} as the "average" over all possible mappings from $\text{scc}(v)$ to $\text{scc}(w)$.

Here are the new polynomial axioms that we have to derive in order to go from $P_{\text{iso}}(G, H)$ to $P_{\text{iso}}(G', H')$:

$$\sum_{v \in V(G') \setminus V(G)} X_{f(vw)} - 1 \qquad \text{for all } w \in V(H') \setminus V(H). \tag{4}$$

$$\sum_{w \in V(H') \setminus V(H)} X_{f(vw)} - 1 \qquad \text{for all } v \in V(G') \setminus V(G). \tag{5}$$

$$X_{f(vw)}X_{v'w'} \qquad \begin{array}{l} \text{for all } v, v' \in V(G'), w, w' \in V(H') \\ \text{such that } (v, w) \in \textbf{NewPairs} \\ \text{and } v', w' \in V(G) \cup V(H), v' \sim w' \\ \text{and } \{(v, w), (v', w')\} \text{ is not} \\ \text{a local isomorphism.} \end{array} \tag{6}$$

$$X_{f(vw)}X_{f(v'w')} \qquad \begin{array}{l} \text{for all } v, v' \in V(G'), w, w' \in V(H') \\ \text{such that } (v, w) \in \textbf{NewPairs} \\ \text{and } (v', w') \in \textbf{NewPairs} \\ \text{and } \{(v, w), (v', w')\} \text{ is not} \\ \text{a local isomorphism.} \end{array} \tag{7}$$

The relation $\sim$ is the same-colour-relation, as in Definition 7. Note that Axioms (4) and (5) only sum over vertices of the same colour as w and v, respectively (as Axioms (1) and (2) do), because $V(G') \setminus V(G)$ and $V(H') \setminus V(H)$ are the sets of newly added vertices. These vertices receive a new colour distinct from all other vertex colours in G and H (see definition of the DWL-operations in Section 7.2).

The next step is to verify that $P_{\text{iso}}(G', H')$ is indeed derivable from $P_{\text{iso}}(G, H)$. This is quite lengthy but does not require any substantial new arguments beyond the lemmas from Section 8. Therefore, we only show the derivation of Axioms (4) and (5) here. The derivation of the other axioms is similar and can be found in the full version of the paper [24].

Derivation of Axioms (4) and (5). Fix $w \in V(H') \setminus V(H)$. We show how to derive Axiom (4) for w. Two cases have to be distinguished, namely whether op = `addPair` or op = `contract`.

Case 1: op = `addPair`: Let $(w_1, w_2) = \text{pair}(w)$. Using the multiplication rule and linear combinations, we derive from Axiom (1) for w_2 in $P_{\text{iso}}(G, H)$:

$$\left(\sum_{\substack{v_1 \in V(G),\\ v_1 \sim w_1}} X_{v_1 w_1}\right) \cdot \left(\sum_{\substack{v_2 \in V(G),\\ v_2 \sim w_2}} X_{v_2 w_2} - 1\right)$$

$$= \sum_{\substack{v_1, v_2 \in V(G)\\ v_1 \sim w_1,\\ v_2 \sim w_2}} X_{v_1 w_1} X_{v_2 w_2} - \sum_{\substack{v_1 \in V(G),\\ v_1 \sim w_1}} X_{v_1 w_1}$$

Recall from the statement of Lemma 22 that $R \in \sigma$ is the colour such that op(R) is executed to obtain $G' \uplus H'$ from $G \uplus H$. Further, let $C = C(G \uplus H)$.
Since $(w_1, w_2) \in R(C)$, we can use Lemma 15 and Lemma 9 to derive from $P_{\text{iso}}(G, H)$ all monomials $X_{v_1 w_1} X_{v_2 w_2}$ where $(v_1, v_2) \notin R(C)$. Hence, we may cancel these monomials from the above sum with the linear combination rule. This yields:

$$\sum_{\substack{v_1, v_2 \in V(G)\\ (v_1, v_2) \in R(C)}} X_{v_1 w_1} X_{v_2 w_2} - \sum_{\substack{v_1 \in V(G)\\ v_1 \sim w_1}} X_{v_1 w_1}.$$

Here, we used that for all pairs $(v_1, v_2) \in V(G)^2 \cap R(C(A))$, it holds that $v_1 \sim w_1$ and $v_2 \sim w_2$. This follows from Corollary 18 and the fact that vertex-colours are represented by diagonal relations. Now we are almost done: We add Axiom (1) for w_1 to the above expression and replace each remaining monomial $X_{v_1 w_1} X_{v_2 w_2}$ with the new extension variable $X_{f(vw)}$, where $v \in V(G')$ is the respective new pair-vertex with $\text{pair}(v) = (v_1, v_2)$. One can see that

$$V(G') \setminus V(G) = \{v \in V(G') \mid \text{pair}(v) \in R(C) \cap V(G)^2\}.$$

Thus, we have indeed derived Axiom (4) for w.
Similarly, we get Axiom (5) for a vertex $v \in V(G') \setminus V(G)$ if we perform the same derivations from the Axioms (2) instead of (1).

Case 2: op = `contract`: We derive Axiom (4) for a fixed vertex $w \in V(H') \setminus V(H)$. Now w is a vertex that represents a contracted R-SCC $\text{scc}(w) \subseteq V(H)$.
For every vertex $w' \in \text{scc}(w)$, we have Axiom (1) for w' in $P_{\text{iso}}(G, H)$:

$$\sum_{\substack{v' \in V(G),\\ v' \sim w'}} X_{v' w'} - 1.$$

Now from this, we may cancel all $X_{v' w'}$ where (v', v') and (w', w') are in distinct diagonal relations in C. This is done again by deriving the respective variables $X_{v' w'}$ with Lemma 15 and Lemma 9. After that step, we have for each $w' \in \text{scc}(w)$:

$$\sum_{\substack{v' \in V(G),\\ \text{there is } v \in V(G') \setminus V(G) \text{ with } v' \in \text{scc}(v)}} X_{v' w'} - 1. \qquad (\star)$$

This holds because the vertex $v' \in V(G)$ has the same diagonal colour as $w' \in \text{scc}(w)$ in the coherent configuration C if and only if it is also contained in some R-SCC (Lemma 19).

Next, we use the variable introduction rule and introduce the variables $X_{f(vw)}$ for every $v \in V(G') \setminus V(G)$. That means, we obtain the following polynomials:

$$\frac{1}{|\mathrm{scc}(v)|} \sum_{(v',w') \in \mathrm{scc}(v) \times \mathrm{scc}(w)} X_{v'w'} - X_{f(vw)} \qquad \text{for each } v \in V(G') \setminus V(G).$$

Now take the sum of all polynomials ($\star$) for all $w' \in \mathrm{scc}(w)$, multiplied by $\frac{1}{|\mathrm{scc}(w)|}$. From this, subtract the above polynomials for all $v \in V(G') \setminus V(G)$. This yields Axiom (4) for the vertex w because we have $\frac{1}{|\mathrm{scc}(v)|} = \frac{1}{|\mathrm{scc}(w)|}$ for all $v \in V(G') \setminus V(G)$, since all R-SCCs have equal size (Lemma 21). In a similar way we can derive Axiom (5) for an SCC-vertex $v \in V(G') \setminus V(G)$.

Due to space limitations, we omit the derivation of Axioms (6) and (7) in this version of the paper. The key idea is again to use that certain vertex pairs receive distinct colours in C and thus, the corresponding monomials can be cancelled with Lemma 15 and Lemma 9. For this, we also need some more properties of coherent configurations, which are not difficult to prove using the correspondence between coherent configurations, 2-WL, and counting logic.

In total, we can derive $P_{\mathrm{iso}}(G', H')$ from $P_{\mathrm{iso}}(G, H)$. The number of new variables is clearly bounded by $|V(G')|^2$. It is also not difficult to see that only polynomially many monomials occur in the derivation, and the binary encoding of the coefficients occurring in them has complexity at most $\mathcal{O}(\log |V(G)|)$ (this holds also in the omitted cases). The used extension axioms are all for polynomials that are averaged sums or degree-2 monomials, as mentioned in Lemma 22. The derivations obtained with Lemma 9 also have polynomial complexity because they can be carried out in MC_3. ◀

10 Discussion and future work

We have shown that the degree-3 extended polynomial calculus can simulate the pair- and contract-operations of Deep Weisfeiler Leman in the sense that the axiom system $P_{\mathrm{iso}}(G', H')$ is derivable from $P_{\mathrm{iso}}(G, H)$ if there is a sequence of DWL-operations that transforms $G \uplus H$ into $G' \uplus H'$. Together with the simulation of k-dimensional Weisfeiler Leman in the degree-k monomial calculus given in [3], this shows that EPC_3 can distinguish two graphs G and H if they can be distinguished in DWL, and the EPC_3-refutation has the same complexity as the DWL-algorithm. Since DWL-algorithms and CPT-programs mutually simulate each other, this result upper-bounds the graph distinguishing power of CPT by that of EPC_3.

This raises the question whether a super-polynomial lower bound for graph isomorphism can be established for EPC_3, preferably for graph classes such as unordered CFI-graphs or multipedes, whose isomorphism problem reduces to a linear equation system and is thus in Ptime. If such a lower bound is found, then by Theorem 2, we would also have that CPT $\neq$ Ptime. This would be a huge step forward in understanding the limitations of symmetry-invariant computation and thus in the quest for a logic for Ptime.

Unfortunately, we do not know how strong the system EPC_3 is, and in particular, if the degree-restriction is a true limitation. It may even be the case that EPC_3 polynomially simulates the *unbounded-degree* extended polynomial calculus. Then it would be as strong as extended Frege because in EPC, the extension variables can encode arbitrary polynomials and thereby arbitrary Boolean circuits. This would make it less useful for proving lower bounds against CPT, as extended Frege lower bounds seem to be out of reach at the moment. However, Theorem 1 also asserts that the simulation of CPT is possible using only extension axioms of a limited form, namely for degree-2 monomials and averaged sums. In this restricted

version of EPC_3, the obvious representation of Boolean circuits as polynomials is no longer possible: The Boolean functions $X \wedge Y, X \vee Y$, and $\neg X$ can naturally be represented as the polynomials $X \cdot Y, X + Y - X \cdot Y$, and $1 - X$. When we represent Boolean circuits using extension variables, then each extension variable corresponds to a gate in the circuit. If the only allowed extension axioms are $\frac{}{X_f - f}$ for $f = X \cdot Y$ or $f = \frac{1}{n}\sum_{i=1}^{n^2} X_i$, then the only gates that we can naturally express are AND-gates (with extension axioms of the first type). Neither NOT-gates nor OR-gates can be simulated (directly) by such extension axioms because this requires sums which are not of the form $\frac{1}{n}\sum X_i$. In particular, these extension axioms cannot be applied to polynomials where variables occur with a negative coefficient. Hence, the corresponding circuits are in some sense *monotone*. This is of course no formal proof that EPC_3 with restricted extension axioms is strictly weaker than extended Frege but at least it rules out the natural simulation of Boolean circuits in EPC_3. In total, the success chances of our suggested approach for CPT lower bounds via proof complexity depend highly on the true power of EPC_3 (with restricted extension axioms), and its relation to unrestricted EPC. Investigating this remains a problem for future work.

Symmetry-invariance of the refutations

Actually, our Theorem 1 could be strengthened more: A simulation of CPT in EPC_3 is even possible in a certain *symmetry-invariant* fragment of EPC_3. However, it seems tricky to give a precise definition of "*symmetric* EPC_3" that is both natural and fits the kind of symmetry we encounter in our CPT-simulation. A neat way to put it would be to say that the set of extension axioms used in a derivation has to be closed under symmetries. With the right definition of "symmetries", this is indeed true for the refutation constructed in Lemma 22. Namely, whenever an extension variable $X_{f(vw)}$ is introduced, where v and w are new pair- or SCC-vertices, then we introduce it for *all* $(v, w) \in V(G') \times V(H')$ that are new. The corresponding polynomials $f(vw)$ consist of variables that refer to the vertices in the respective pairs or SCCs of v and w. The automorphisms of the graphs G and H preserve the colours of all vertex-pairs in the coarsest coherent configuration. Therefore, the set of extension axioms that we introduce in each step of the refutation is closed under the automorphisms of G and H. The action of these automorphism groups on the set of variables of $P_{\mathrm{iso}}(G, H)$ is the natural one, i.e. if π is an automorphism of G and σ an automorphism of H, then they take X_{vw} to $X_{\pi(v)\sigma(w)}$. This extends naturally to the extension axioms, so for example, if v and w are pair-vertices with $\mathrm{pair}(v) = (v_1, v_2), \mathrm{pair}(w) = (w_1, w_2)$, then the extension axiom $X_{f(vw)} - X_{v_1w_1}X_{v_2w_2}$ is mapped to $X_{f(v'w')} - X_{\pi(v_1)\sigma(w_1)}X_{\pi(v_2)\sigma(w_2)}$, where v', w' are the newly introduced pair-vertices for $(\pi(v_1), \pi(v_2))$ and $(\sigma(w_1), \sigma(w_2))$ (such v', w' must exist because DWL is isomorphism-invariant and introduces new vertices for all pairs with the same colour). So in this sense, the extension axioms used in Lemma 22 are closed under all automorphism-pairs $(\pi, \sigma) \in \mathbf{Aut}(G) \times \mathbf{Aut}(H)$.
Unfortunately, this does not lead to a *general* definition of symmetric EPC_3 because it depends on the automorphisms of G and H, the graphs which are implicitly encoded in $P_{\mathrm{iso}}(G, H)$. When EPC_3 is applied to other polynomial axiom systems, then there might be no graphs "in the background". So for a general set of input polynomials $\mathcal{P}$, it would be natural to require that the set of extension axioms in a refutation be closed under the automorphisms of $\mathcal{P}$ – these are the permutations of the variables that extend to permutations of the polynomials in $\mathcal{P}$. However, this would no longer fit to our derivation from Lemma 22: The system $P_{\mathrm{iso}}(G, H)$ in general has more automorphisms than $\mathbf{Aut}(G) \times \mathbf{Aut}(H)$. Namely, $P_{\mathrm{iso}}(G, H)$ contains no information about where the edges and non-edges in G and H actually are; it just relates pairs $(v, v') \in V(G)^2$ with pairs $(w, w') \in V(H)^2$ where (v, v') is an edge

and (w, w') is not, or vice versa (Axiom (3)). Therefore, an automorphism of $P_{\text{iso}}(G, H)$ may swap all edges with non-edges, as long as it does so in both G and H (such examples can be constructed). But the automorphisms of the graphs must preserve edges and non-edges, so such an automorphism of $P_{\text{iso}}(G, H)$ does not correspond to one from $\mathbf{Aut}(G) \times \mathbf{Aut}(H)$. Our constructed refutation is only symmetric with respect to the latter. Thus, our simulation of CPT in EPC_3 is possible in a way that respects *specific* symmetries of $P_{\text{iso}}(G, H)$, but we do not know if this kind of symmetry-invariance can be formulated independently of the graph isomorphism problem as a general restriction to the proof system EPC_3. Perhaps future research will reveal a more generic way to define symmetrized versions of known proof systems. This could be of independent interest because it might be possible to prove lower bounds for symmetric versions of proof systems for which non-symmetric lower bounds seem to be out of reach.

Finally, another question that we have not answered is whether the converse to Theorem 1 also holds: Is there an algorithm that can find EPC_3-refutations (for graph isomorphism) and can be implemented in CPT? Since CPT is symmetry-invariant and EPC_3 is not, this seems unlikely. Furthermore, such a proof search algorithm would probably have to be non-deterministic. Thus, the only way to get an exact match in expressive power between the logic and the proof system might be by restricting EPC_3 to a symmetry-invariant fragment and extending CPT with some kind of non-determinism.

References

1 Faried Abu Zaid, Erich Grädel, Martin Grohe, and Wied Pakusa. Choiceless Polynomial Time on structures with small Abelian colour classes. In *Mathematical Foundations of Computer Science 2014*, volume 8634 of *Lecture Notes in Computer Science*, pages 50–62. Springer, 2014. URL: `http://logic.rwth-aachen.de/pub/pakusa/cptcan.pdf`.

2 Yaroslav Alekseev. A Lower Bound for Polynomial Calculus with Extension Rule. In Valentine Kabanets, editor, *36th Computational Complexity Conference (CCC 2021)*, volume 200 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 21:1–21:18, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. `doi:10.4230/LIPIcs.CCC.2021.21`.

3 Christoph Berkholz and Martin Grohe. Limitations of algebraic approaches to graph isomorphism testing. In *International Colloquium on Automata, Languages, and Programming*, pages 155–166. Springer, 2015.

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

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

6 Ashok K 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`.

7 Gang Chen and Ilia Ponomarenko. Lectures on coherent configurations. *Lecture notes*, 2019. Available at `http://www.pdmi.ras.ru/~inp/ccNOTES.pdf`.

8 Matthew Clegg, Jeffery Edmonds, and Russell Impagliazzo. Using the Groebner basis algorithm to find proofs of unsatisfiability. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 174–183, 1996.

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

10 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.

11 Susanna F de Rezende, Massimo Lauria, Jakob Nordström, and Dmitry Sokolov. The power of negative reasoning. In *36th Computational Complexity Conference (CCC 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

12 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.

13 Erich Grädel and Martin Grohe. Is polynomial time choiceless? In *Fields of Logic and Computation II*, pages 193–209. Springer, 2015.

14 Erich Grädel, Martin Grohe, Benedikt Pago, and Wied Pakusa. A finite-model-theoretic view on propositional proof complexity. *Logical Methods in Computer Science*, 15, 2019.

15 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.

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

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

18 Yuri Gurevich and Saharon Shelah. On finite rigid structures. *The Journal of Symbolic Logic*, 61(2):549–562, 1996.

19 Lauri Hella. Logical hierarchies in PTIME. *Information and Computation*, 129(1):1–19, 1996.

20 Neil Immerman and Eric Lander. Describing graphs: A first-order approach to graph canonization. In *Complexity theory retrospective*, pages 59–81. Springer, 1990.

21 Sandra Kiefer. The Weisfeiler-Leman algorithm: an exploration of its power. *ACM SIGLOG News*, 7(3):5–27, 2020.

22 Moritz Lichter. Separating rank logic from polynomial time. *CoRR*, abs/2104.12999, 2021. arXiv:2104.12999.

23 Benedikt Pago. Choiceless Computation and Symmetry: Limitations of Definability. In Christel Baier and Jean Goubault-Larrecq, editors, *29th EACSL Annual Conference on Computer Science Logic (CSL 2021)*, volume 183 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 33:1–33:21, Dagstuhl, Germany, 2021. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CSL.2021.33.

24 Benedikt Pago. Finite model theory and proof complexity revisited: Distinguishing graphs in Choiceless Polynomial Time and the Extended Polynomial Calculus. *arXiv*, 2022. arXiv:2206.05086.

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

26 Benjamin Rossman. Choiceless computation and symmetry. In *Fields of logic and computation*, pages 565–580. Springer, 2010.

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