Is Polynomial Time Choiceless?

Erich Grädel and Martin Grohe

RWTH Aachen University, Germany
graedel@logic.rwth-aachen.de,
grohe@informatik.rwth-aachen.de

For Yuri Gurevich on the occasion of his 75th birthday

Abstract. A long time ago, Yuri Gurevich made precise the problem of whether there is a logic capturing polynomial-time on arbitrary finite structures, and conjectured that no such logic exists. This conjecture is still open. Nevertheless, together with Andreas Blass and Saharon Shelah, he has also proposed what still seems to be the most promising candidate for a logic for polynomial time, namely Choiceless Polynomial Time (with counting). We survey some recent results on this logic.

1 Introduction

Is there a logic for PTIME? More than thirty years after this problem has first been posed by Chandra and Harel (in somewhat different form) in the context of database theory, and after Yuri Gurevich has reformulated the question in logical terms, we still do not know the answer. The quest for a logic for PTIME, or for a proof that no such logic exists, still remains the perhaps most fundamental and challenging open problem of finite model theory.

Yuri Gurevich has made many important contributions to this problem, including numerous studies on the expressive power, structure, and complexity of a number of different logics that arise in this context. Let us focus here on two main achievements.

The precise formulation of what would really constitute a logic for PTIME has been extremely influential, and is of course an indispensable prerequisite for all attempts to prove that such a logic cannot exist. Gurevich's first requirement is that a *logic* should have a decidable syntax, that is, a decidable set of of *sentences*. Each sentence *defines* a *property* of finite[1] structures, that is, an isomorphism closed class of structures of the same vocabulary. For the logic to *capture* polynomial time we want every polynomial-time decidable property of structures to be definable by some sentence of the logic. This is Gurevich's second requirement for a logic capturing PTIME. Conversely, we want every property definable in the logic to be decidable in polynomial time. However, this is not sufficient to exclude pathological examples. For example, we could take an (arbitrary, not necessarily

[1] Structures are always assumed to be finite in this paper, with the exception of the herditarily finite expansions introduced in Section 2.

computable) enumeration $P_0, P_1, P_2, \ldots$ of all polynomial-time decidable properties of structures and define a logic whose sentences are the natural numbers and where sentence i defines the property P_i. What we are still missing is an effective link between sentences and the properties they define. Therefore, the third requirement for a logic capturing PTIME is the existence of a "compiler" that translates a sentence into a polynomial time evaluation algorithm, that is, an algorithm that computes for each sentence φ of the logic a polynomial time algorithm A_φ deciding the property P_φ defined by φ.

Despite his conjecture that there is no logic capturing PTIME, Yuri Gurevich has, together with Andreas Blass and Saharon Shelah [6], also proposed *Choiceless Polynomial Time*, a logical formalism that, arguably, is still the most promising candidate for a logic that might actually capture PTIME. This paper is a survey of recent results on Choiceless Polynomial Time.

2 Choiceless Polynomial Time

There are several different presentations of Choiceless Polynomial Time (CPT). The original intention was to explore a model for efficient computations on abstract finite structures (and not on presentations of these via finite strings) which preserve symmetries at every step in the computation. This prohibits the explicit introduction of an ordering or, equivalently, arbitrary choices between indistinguishable elements of the input structure (or of the current state). Notice that such choices appear in many algorithms of fundamental importance, including depth-first search, Gaussian elimination, the augmenting-path algorithm for bipartite matching and many more.

Thus, Blass, Gurevich, and Shelah set out to define a computation model that avoids symmetry breaking choices, but allows essentially everything else, including parallelism and "fancy data structures", as long as all operations can be carried out in polynomial time. For a precise definition, they proposed a model based on abstract state machines, which, given a finite input structure, works on its extension by all hereditarily finite sets over it, which may be seen as a powerful higher-order data structure.

Inspired by Rossman [27], we give a more "logical" definition of CPT.

2.1 BGS-Logic

We need to review a few set theoretic notions first. For a finite set A, the set $\mathrm{HF}(A)$ of *hereditarily finite sets* over A is defined as follows: we let $H_0 := A \cup \{\emptyset\}$ and $H_{i+1} := H_i \cup 2^{H_i}$ for all $i \geq 0$ and $\mathrm{HF}(A) := \bigcup_{i \geq 0} H_i$. For every $x \in \mathrm{HF}(A)$, we define the *rank* of x to be the least i such that $x \in H_i$. We call the elements $a \in A \subseteq \mathrm{HF}(A)$ the *atoms*; all other elements of $\mathrm{HF}(A)$ are *sets*. With every natural number $n \in \mathbb{N}$ we associate the corresponding *van Neumann ordinal* n, inductively by $\mathsf{0} := \emptyset$ and $\mathsf{n{+}1} := \mathsf{n} \cup \{\mathsf{n}\}$.

Suppose now that we have a finite structure $\mathfrak{A}$ with universe A over some vocabulary τ consisting of relation, function, and constant symbols. We let τ^{HF}

be the extension of τ by a binary relation symbol $\in$, a binary function symbol Pair, unary function symbols Union, TheUnique, Card, and constant symbols $\emptyset$ and Atoms (of course we assume that none of these symbols appears in τ). We define the *hereditarily finite expansion of* A to be the τ^{HF}-structure $\mathrm{HF}(\mathfrak{A})$ with universe $\mathrm{HF}(A)$, all symbols from τ interpreted as in $\mathfrak{A}$, with the convention that functions take all arguments not from A to $\emptyset$, and the new symbols in $\tau^{\mathrm{HF}} \setminus \tau$ are interpreted in the natural way: $\in$ is the binary "element"-relation, Pair maps x, y to the set $\{x, y\}$, Union maps x to the union of all sets in x, TheUnique maps singleton sets $\{x\}$ to their unique element x and all other sets to the empty set, Card maps a set to its cardinality (represented as a von Neumann ordinal), $\emptyset$ is the empty set, and Atoms is the set A, viewed as an element of $\mathrm{HF}(A)$ of rank 1.

The next step towards the definition of CPT is the definition of a more general logic called *BGS-logic*. The syntactic objects of BGS-logic are *terms* and *formulae*. There are three different types of terms, or rather, term constructions: *ordinary terms*, *comprehension terms*, and *iteration terms*. The *ordinary terms* of BGS-logic of vocabulary τ are τ^{HF}-terms defined in the usual way. The *formulae* of BGS-logic are Boolean combinations of *atomic formulae* of the form $t = u$, $t \in u$, and $R(t_1, \ldots, t_k)$ for every k-ary relation symbol $R \in \sigma$, where $t, u, t_1, \ldots, t_k$ are terms. Thus all formulae are quantifier-free. (Example 2 below gives an indication on how quantifiers can be simulated.) Besides the ordinary terms, BGS-logic has *comprehension terms* of the form

$$\{t\colon v \in u\colon \varphi\}, \tag{1}$$

where t, u are terms, v is a variable that is not free in u, and φ is a formula. More suggestively, we may write $\{t(v)\colon v \in u\colon \varphi(v)\}$ to indicate that this term defines the set of all values $t(x)$, where x is an element of the set defined by the term u that satisfies the formula φ. Note that t, u, and v may have other free variables besides v.

Finally, for every term t with just one free variable we have an *iteration term* t^*. More suggestively, we may write $t(v)^*$, where v is the free variable of t. Intuitively, the value of the term t^* is the first fixed point of the sequence $t(\emptyset), t(t(\emptyset), t(t(t(\emptyset))), \ldots$, if such a fixed point exists, or $\emptyset$ if no fixed point exists.

We define the *free variables* of terms and formulae in the natural way, stipulating that the variable v in a comprehension term of the form (1) and in an iteration term $t(v)^*$ be bound. Thus iteration terms t^* can never have free variables.

Terms and formulae of vocabulary τ are interpreted in the hereditarily finite expansion of τ-structures in the natural way. Formally, the denotation of a term $t = t(v_1, \ldots, v_k)$ of vocabulary τ and with free variables $v_1, \ldots, v_k$ in a τ-structure $\mathfrak{A}$ is the function $[\![t]\!]^{\mathfrak{A}} : \mathrm{HF}(A)^k \to \mathrm{HF}(A)$ that maps $(x_1, \ldots, x_k) \in \mathrm{HF}(A)^k$ to the value of the term if $v_1, \ldots, v_k$ are interpreted by $x_1, \ldots, x_k$, repectively. If a term t has no free variables, then $[\![t]\!]^{\mathfrak{A}}$ is a nullary function, which we interpret as a constant. Similarly, we define the denotation $[\![\varphi]\!]^{\mathfrak{A}}$ of a formula $\varphi = \varphi(v_1, \ldots, v_k)$ in $\mathfrak{A}$ to be the set of all $(x_1, \ldots, x_k) \in \mathrm{HF}(A)^k$ satisfying φ. Note that if φ is a *sentence*, that is, a formula without free variables, then

either $[\![\varphi]\!]^{\mathfrak{A}} = \{\emptyset\} =: \mathsf{True}$ or $[\![\varphi]\!]^{\mathfrak{A}} = \emptyset =: \mathsf{False}$. In these and similar notations, we omit the superscript$^{\mathfrak{A}}$ if $\mathfrak{A}$ is clear from the context. The definition is straightforward for ordinary terms and formulae. For a comprevension term $s := \{t\colon v \in u\colon \varphi\}$, where for simplicity we assume that t and φ have the same free variables $v, v_1, \dots, v_k$ and u has free variables $v_1, \dots, v_k$, we define

$$[\![s]\!](x_1, \dots, x_k) := \big\{ [\![t]\!](x, x_1, \dots, x_k) \;\big|\; x \in [\![u]\!](x_1, \dots, x_k) \text{ such that } (x, x_1, \dots, x_k) \in [\![\varphi]\!] \big\},$$

for all $(x_1, \dots, x_k) \in \mathrm{HF}(A)^k$. For an iteration term t^* we define a sequence $(x_i)_{i\geq 0}$ by $x_0 := \emptyset$ and $x_{i+1} := [\![t]\!](x_i)$, and we let

$$[\![t^*]\!] := \begin{cases} x_\ell & \text{for the least } \ell \text{ such that } x_\ell = x_{\ell+1} \text{ if such an } \ell \text{ exists,} \\ \emptyset & \text{otherwise.} \end{cases}$$

If there is no ℓ such that $x_\ell = x_{\ell+1}$, we say that t^* *diverges* in $\mathfrak{A}$. We define the *length of the iteration* $\mathrm{len}(t^*, \mathfrak{A})$ of t^* in $\mathfrak{A}$ to be the least ℓ such that $x_\ell = x_{\ell+1}$, or ∞ if t^* diverges.

Remark 1. Instead of defining $[\![t^*]\!]$ to be the emptyset if t^* diverges, we could also leave it undefined and work with a three valued logic. This would be closer to Blass, Gurevich and Shelah's original approach, but complicate things unnecessarily. For the fragment CPT of BGS-logic that we are mainly interested in, this makes no difference, because all terms in this fragment will be required to converge anyway.

The iteration terms play the role of the *programs* in Rossman's version of BGS-logic. It is easy to see that programs can be simulated by iteration terms and, conversely, iteration terms can be simulated by programs. However, as opposed to Rossman, we allow iteration terms to appear inside of other terms and formulae, whereas Rossman does not allow nested programs. But again, for the fragment CPT this makes no difference.

Example 2 (Triangles in a Graph). Let $\tau = \{E\}$ with one binary relation symbol E; we view τ-structures as directed, or if E is symmetric undirected, graphs. We shall construct a BGS-term that defines the set of all triangles (viewed as 3-element sets) in an undirected graph.

We let $\varphi(v_1, v_3, v_3) := E(v_1, v_2) \wedge E(v_2, v_3) \wedge E(v_3, v_1)$ and

$$\begin{aligned} t_1(v_2, v_3) &:= \{\mathsf{Union}(\mathsf{Pair}(\mathsf{Pair}(v_1, v_1), \mathsf{Pair}(v_2, v_3)))\colon v_1 \in \mathsf{Atoms}\colon \varphi(v_1, v_3, v_3)\}, \\ t_2(v_3) &:= \{t_1(v_2, v_3)\colon v_2 \in \mathsf{Atoms}\colon v_2 = v_2\}, \\ t_3 &:= \{t_2(v_3)\colon v_3 \in \mathsf{Atoms}\colon v_3 = v_3\}. \end{aligned}$$

Then $[\![t_3]\!]^{\mathfrak{G}}$ is the set of all triangles of an undirected graph $\mathfrak{G}$.

We can now define a formula $\psi(v)$ expressing that v is contained in a triangle: we simply let $\psi(v) := v \in \mathsf{Union}(t_3)$.

Using ideas similar to those in the example, it is not difficult to express bounded quantifiers $\exists v \in t$ and $\forall v \in t$ in BGS-logic. Thus all formulae of bounded first-order logic (called Δ_0-formulae in set theory) over the hereditarily finite sets can be expressed by equivalent BGS-formulae (without using iteration).

Example 3 (Power Set). We shall construct a BGS-term that defines the power set of the set of atoms.

We observe that we can express the union of two sets s and t as

$$s \cup t := \mathsf{Union}(\mathsf{Pair}(s, t)).$$

We define an auxiliary term

$$s(w) := \{w \cup \mathsf{Pair}(w', w') \colon w' \in \mathsf{Atoms} \colon w' = w'\},$$

which defines the collection of sets obtained by adding an atom to w, and let

$$t(v) := \mathsf{Pair}(\emptyset, \emptyset) \cup \mathsf{Union}(\{s(w) \colon w \in v \colon w = w\}).$$

Then $\overbrace{t(t(\cdots t}^{i \text{ times}}(\emptyset)\cdots))$ defines the set of sets of at most $i-1$ atoms. Thus for every structure $\mathfrak{A}$ we have $[\![t^*]\!] = 2^A$.

We can use the previous example to show that BGS-logic can simulate monadic second-order logic, and by iterating the construction, higher order logic. Thus the logic is far too powerful to stay within the realm of polynomial time computations.

2.2 Definition of Choiceless Polynomial Time

Intuitively, *Choiceless Polynomial Time* (CPT) is the polynomial-time fragment of BGS-logic. To define CPT we first restrict the length of iterations to be polynomial in the size of the input. However, this alone is not sufficient, as can be seen by Example 3. In addition, it is necessary to restrict also the number of elements that are being used, or *active*, in any step of the computation.

We inductively define for every term $s = s(v_1, \ldots, v_k)$, every structure $\mathfrak{A}$, and every tuple $\overline{x} = (x_1, \ldots, x_k) \in \mathrm{HF}(A)^k$ the set $\mathrm{act}(s, \mathfrak{A}, \overline{x})$ of *active elements* as follows.

- If $s = v$ is a variable then $\mathrm{act}(s, \mathfrak{A}, x) = \{x\}$, and if $s = c$ is a constant then $\mathrm{act}(s, \mathfrak{A}) = \{[\![c]\!]\}$.
- If $s = f(t_1, \ldots, t_\ell)$ then

$$\mathrm{act}(s, \mathfrak{A}, \overline{x}) = \{[\![s]\!](\overline{x})\} \cup \bigcup_{i=1}^{\ell} \mathrm{act}(t_i, \mathfrak{A}, \overline{x}_i),$$

  where $\overline{x}_i$ is the subtuple of $\overline{x}$ corresponding the free variables of t_i.

– If $s = \{t\colon v \in u\colon \varphi(v)\}$, then

$$\operatorname{act}(s, \mathfrak{A}, \overline{x}) = \{[\![s]\!](\overline{x})\} \cup \operatorname{act}(u, \mathfrak{A}, \overline{x}) \cup \bigcup_{x \in [\![u]\!](\overline{x})} \big(\operatorname{act}(t, \mathfrak{A}, x\overline{x}) \cup \operatorname{act}(\varphi, \mathfrak{A}, x\overline{x})\big),$$

where $x\overline{x}$ denotes the tuple $(x, x_1, \dots, x_k)$ and for simplicity we assume that the free variables of t and φ are $v, v_1, \dots, v_k$ and the free variables of u are $v_1, \dots, v_k$.

– If $s = t^*$ we define the sequence $(x_i)_{i \geq 0}$ as in the definition of $[\![t^*]\!]$ and let

$$\operatorname{act}(s, \mathfrak{A}) = \bigcup_{i \geq 0} \operatorname{act}(t, \mathfrak{A}, x_i).$$

Note that if s converges then $\operatorname{act}(s, \mathfrak{A}) = \bigcup_{i=0}^{\operatorname{len}(t^*, \mathfrak{A})} \operatorname{act}(t, \mathfrak{A}, x_i)$, and thus $\operatorname{act}(s, \mathfrak{A}) \in \operatorname{HF}(A)$. This is not necessarily the case if t diverges.

For a formula φ, we define $\operatorname{act}(\varphi, \mathfrak{A}, \overline{x})$ to be the union of the sets $\operatorname{act}(t, \mathfrak{A}, \overline{x}_t)$ for the terms t that are used to built φ.

We are now ready to define *Choiceless Polynomial Time (CPT)* as the fragment of BGS-logic consisting of all sentences φ for which there is a polynomial $p(x)$ such that for all structures $\mathfrak{A}$ of order $|A| = n$ we have

$$|\operatorname{act}(\varphi, \mathfrak{A})| \leq p(n).$$

Observe that this implies that for all iteration terms t^* appearing in φ (also as subterms of other terms) we have $\operatorname{len}(t^*, \mathfrak{A}) \leq p(n)$, because all values appearing in the steps of the iteration are active elements.

Remark 4. Defined this way, CPT is not a logic in the strict sense because it does not have a decidable syntax. However, it is easy to define a syntactic fragment that does have a decidable syntax, but still has the same expressive power. The key idea is to include explicit counters and cardinality tests in iteration terms. The next example illustrates how this can be done.

Example 5 (Counter). In this example, we show how to modify an iteration term s^* in such a way that the iteration is aborted if no fixed-point is reached after n (= order of the structure) steps.

It is not hard to define a binary term $\langle \cdot, \cdot \rangle$ that combines its two arguments into an ordered pair and projection terms π_1, π_2 that map an ordered pair to its entries. Moreover, the term $\mathsf{succ}(v) := v \cup \mathsf{Pair}(v)$ maps a von-Neumann ordinal n to ist successor $\mathsf{n{+}1}$. Note that $\mathsf{m} < \mathsf{n} \;\Leftrightarrow\; \mathsf{m} \in \mathsf{n}$ for all $n, m \in \mathbb{N}$.

Now consider an iteration term s^*. We let

$$t(v) := \langle \mathsf{0}, \emptyset \rangle \cup \{\langle \mathsf{succ}(\pi_1(w)), s(\pi_2(w)) \rangle\colon w \in v\colon \pi_1(w) \in \mathsf{Card}(\mathsf{Atoms})\}.$$

Then for all structure $\mathfrak{A}$ of order $n := |A| = [\![\mathsf{Card}(\mathsf{Atoms})]\!]$ we have $[\![t^*]\!] = \{(\mathsf{i}, x_i) \mid 0 \leq i \leq n\}$, where the sequences $(x_i)_{i \geq 0}$ is defined by $x_0 := \emptyset$ and

$x_{i+1} := [\![s]\!](x_i)$. Obsserve that if s^* converges then $x_n = s^*$ if and only if $x_n = x_{n+1}$. Let

$$u := \mathsf{TheUnique}\Big(\big\{\pi_2(w) \colon w \in t^* \colon \pi_1(w) = \mathsf{Card}(\mathsf{Atoms}) \wedge \pi_2(w) = s(\pi_2(w))\big\}\Big).$$

Then $[\![u]\!] = [\![s^*]\!]$ if $\mathrm{len}(s, \mathfrak{A}) \leq n$ and $[\![u]\!] = \emptyset$ otherwise.

2.3 Defining Properties of Small Substructures

To illustrate the power of CPT, we consider specific structures that we call padded graphs. The vocabulary τ consist of a unary relation symbol V and a binary relation symbol E. A *padded graph* is a τ-structure $\mathfrak{A}$ where $E^{\mathfrak{A}} \subseteq V^{\mathfrak{A}} \times V^{\mathfrak{A}}$ and $E^{\mathfrak{A}}$ is symmetric and irreflexive. The *underlying graph* of a padded graph $\mathfrak{A}$ is the graph $\mathfrak{G}_{\mathfrak{A}}$ with vertex set $V^{\mathfrak{A}}$ and edge relation $E^{\mathfrak{A}}$. In the following, we always use n to denote the order $|A|$ of a padded graph $\mathfrak{A}$ and ℓ to denote the order $|V^{\mathfrak{A}}|$ of its underlying graph. We usually assume that $\ell \ll n$.

Example 6 (3-Colourability of Padded Graphs). In this example, we consider padded graphs $\mathfrak{A}$ where $\ell \leq \log n$. Using the construction of Example 3, we obtain a term t that defines the powerset of V, that is, $[\![t]\!] = 2^{V^{\mathfrak{A}}}$. The assumption $\ell \leq \log n$ guarantees that this works within the polynomial bounds imposed by CPT. Using this term, we can easily write a CPT-sentence of the form $\exists v_1 \in t\, \exists v_2 \in t\, \exists v_3 \in t\, (\cdots)$ stating that the underlying graph of $\mathfrak{A}$ is 3-colourable.

The following example is due to Blass, Gurevich, and Shelah.

Example 7 (Linear Orders). We consider padded graphs where $\ell! \leq n$. Then, using a similar idea as in Examples 3 and 6, we construct a term t that defines the set of all linear orders of V. As there are $\ell!$ linear orders of V, the assumption $\ell! \leq n$ guarantees that we stay within the polynomial bounds imposed by CPT.

Now, exploiting the facts that least fixed-point logic LFP captures polynomial time on ordered graphs and that CPT is at least as expressive as LFP, it is easy to show that CPT can express all polnomial time properties of the underlying graph of the given padded graph.

It is not hard to show that there are polynomial time properties of the underlying graph that cannot be expressed in LFP, not even in fixed-point logic with counting FPC (see Section 3), because the padding does not help these logics very much. This is the easiest way to show that CPT is strictly more expressive than FPC.

Laubner [25], in his PhD-thesis, slightly strengthened the result of the previous example and proved the following theorem, which intuitively says that CPT expresses all polynomial time properties of definable subgraphs of logarithmic size.

Theorem 8. *For every property P of graphs that is decidable in polynomial time there is a CPT-sentence φ such that for all padded graphs $\mathfrak{A}$ with $\ell \leq \log n$, the following are equivalent.*

1. $\mathfrak{A}$ *satisfies* φ.
2. *The underlying graph* $\mathfrak{G}_{\mathfrak{A}}$ *has property* P.

The crucial step in the proof of this theorem is the implementation of a combinatorial graph canonization algorithm due to Corneil and Goldberg [11], running in time $2^{O(\ell)}$ on graphs of order ℓ, in CPT.

2.4 Choiceless Polynomial Time Without Counting

To be a serious candidate for being a logic for polynomial time, CPT has (and needs) the cardinality operator Card. Blass, Gurevich and Shelah also considered a variant of CPT without the Card-operator, which we denote by CPT^-.

Not surprisingly, CPT^- is unable to determine whether a structure has an even or odd number of elements, but this is much more difficult to prove than for, say, least fixed-point logic. The proof requires a sophisticated analysis of the support of hereditarily finite sets used in CPT-computations (see [27]). Nevertheless CPT^- is quite a powerful language; for instance it has been in shown in [14] that CPT^- can express (a variant of) the CFI-query that separates fixed-point logic with counting (FPC) from PTIME and is therefore incomparable with FPC. An interesting result on CPT^- is the zero-one law established by Shelah (see [5] for details) saying that for every CPT^--definable property P of relational τ-structures the probability $\mu_n(P)$ that a random τ-structure of cardinality n satisfies property P tends either to 0 or 1 as n goes to infinity.

3 Fixed-Point Logic With Counting

The logic of reference, or yardstick, in the search for a logic for PTIME is fixed-point logic with counting, denoted FPC. This logic was introduced, somewhat informally, by Immerman [23], a more formal definition, based on two-sorted structures, inflationary fixed-points, and counting terms was given in [17]. For a recent survey on FPC, see [12].

Fixed-point logic with counting comes actually rather close to being a logic for polynomial time. It is strong enough to express most of the fundamental algorithmic techniques leading to polynomial-time procedures and it captures PTIME on many interesting classes of finite structures, including trees, planar graphs, structures of bounded tree width, and actually all classes of graphs with an excluded minor [20]. Indeed, these classes even admit FPC-*definable canonisation* which means that FPC can define, given an input structure, an isomorphic copy of that structure over a linearly ordered universe. Clearly, if a class of structures admits FPC-definable canonisations, then FPC captures PTIME on this class, since by the Immerman-Vardi Theorem (see e.g. [19]) fixed-point logic can define every polynomial-time query on ordered structures.

Although it has been known for more than twenty years that FPC fails to capture PTIME in general, by the fundamental CFI-construction due to Cai, Fürer, and Immerman [10], we still know only relatively few properties of finite structures that provably separate FPC from PTIME.

Roughly we have, at this time, two main sources for such problems. The first one includes tractable cases of the isomorphism problem for finite structures, in particular for graphs. It is, in general still open, whether the general graph isomorphism problem is solvable in polynomial time, but efficient isomorphism tests are known in many special cases, including all classes of graphs of bounded degree *or* bounded colour class size. However, the CFI-construction shows that FPC cannot define the isomorphism problem even on graphs with bounded degree *and* bounded colour class size.

Multipedes. An interesting instance of such a problem is the isomorphism problem for *multipedes*. Multipedes[2] have been introduced in [7] and studied also in [21]. Informally, a multipede is a finite two-sorted structure, consisting of an ordered set of segments, and a set of feet, such that exactly two feet are attached to each segment. Further there is a collection of hyperedges H of size 3 on the segments, and a corresponding collection of hyperedges P of feet, also of size 3, called positive triples, such that each positive triple of feet is attached to a hyperedge H of segments, and out of the eight triples of feet attached to H, exactly four are positive. Further if P and P' are two positive triples of feet attached to the same hyperedge H, then $|P - P'|$ is even. Finally, exactly one of the two feet attached to the first segment carries a shoe.

Blass, Gurevich, and Shelah [7] proved that the isomorphism problem for multipedes can be solved in polynomial time, but that it is not expressible in fixed-point logic with counting. They asked the question whether it is definable in CPT.

Linear algebra. The second class of hard problems for FPC includes queries from linear algebra. In general, the definability of central problems of linear algebra provides an interesting challenge in the study of the expressive power of logical systems and for the quest for a logic for PTIME. On one side, it has turned out that a fair amount of linear algebra, in particular for fields of characteristic zero, is expressible in fixed-point logic with counting, including arithmetic operations on matrices, singularity of matrices, determinants, characteristic polynomials, and matrix rank over $\mathbb{Q}$ (but not over fields of prime characteristic). On the other side, Atserias, Bulatov and Dawar [3] proved that FPC cannot express the solvability of linear equation systems over any finite Abelian group, and it then follows that also a number of other problems from linear algebra are not definable in FPC either. This motivated the introduction of *rank logic*, which extends FPC by operators for the rank of definable matrices over prime fields $\mathbb{F}_p$, and which permits to express the solvability of linear equation systems over finite fields [13]. Interestingly, also the CFI-query can be formulated as linear equation system over $\mathbb{F}_2$ and is thus expressible in rank logic. There are different variants of rank logic. For the most powerful of them, with a rank operator where the prime over which the rank is computed is not fixed, but part of the input, it is still open whether it captures PTIME [18].

[2] Actually, Gurevich and Shelah introduced a number of different variants of multipedes. What we use here are called 3-multipedes with shoes in [7].

4 Structures of Bounded Colour Class Size

Recall a that a preorder of width q is a reflexive, transitive, and total binary relation $\preceq$ such that the induced equivalence relation $a \sim b := (a \preceq b \preceq a)$ only has equivalence classes of size $\leq q$. A *q-bounded structure* is a structure that is equipped with a pre-order $\preceq$ of width q. The equivalence classes induced by $\preceq$ are also called colour classes.

It is still open whether Choiceless Polynomial Time captures PTIME on all classes of finite structures with bounded colour class size. A partial positive answer was given in [1], for any class of *q-bounded structures with Abelian colours*, which means that the automorphism groups of all substructures induced by the colour classes are commutative.

An important ingredient in the CPT-canonization procedure for such classes is a choiceless algorithm for solving a special class of linear equation systems. Clearly linear equation systems over an *ordered* set of variables can be solved in fixed-point logic with counting. However, classical solution algorithms for linear equation systems require choice, and for *unordered* sets of variables they cannot be carried out in FPC. An intermediate class are *cyclic linear equation systems* (CES) over finite rings $\mathbb{Z}_{p^k}$, equipped with a pre-order $\preceq$ on the set of variables, such that every pair of $\preceq$-equivalent variables x, y is related by an equation $x + a = y$ for some constant a. This means that fixing the value of one variable in a solution of the CES fixes also the values for all other variables in the same $\preceq$-class. Cyclic equation systems arise for instance in Cai-Fürer-Immerman (CFI) constructions. The original CFI-query (over ordered input graphs) can be formulated as cyclic equation systems over $\mathbb{Z}_2$ where the cyclic constraint on pairs x_0, x_1 of $\preceq$-equivalent variables simply has the form $x_0 + x_1 = 1$. In Holm's PhD-thesis [22] and also in [18] a generalized CFI-construction over rings $\mathbb{Z}_q$ has been exhibited which is, for instance, relevant for the study of rank logic. Again, the isomorphism problem for generalized CFI-structures, which can be formulated as a CES over $\mathbb{Z}_q$, separates PTIME from FPC, but it also gives rise to a number of further separation results, concerning for instance different variants of rank logics [18].

Theorem 9. *The solvability problem for cyclic linear equation systems can be defined in Choiceless Polynomial Time.*

Any $\preceq$-class of variables in a CES has a cyclic structure, and we can order each such class by fixing one variable. However, in Choiceless Polynomial Time it is not possible to simultaneously fix one variable in each class, since this would require to take into account also all symmetric choices of which there may be exponentially many. One can circumvent this problem by means of so-called *hyperterms* which avoid this exponential blow-up by identifying equivalent choices and encoding equivalence classes as hereditarily finite sets over the universe of variables. Choiceless Polynomial Time is powerful enough to perform arithmetic operation on hyperterms, and to translate any cyclic linear equation system into an *ordered system of hyperequations*. Finally the solvability of such systems can then be determined in CPT by a variant of Gaussian elimination for finite rings.

Cyclic linear equation systems are an essential ingredient in the canonization procedure for q-bounded structures with Abelian colours. For details, we refer to [1] and the forthcoming PhD thesis of Wied Pakusa.

Theorem 10. CPT *captures* PTIME *on every class of q-bounded structures with Abelian colours.*

Notice that 2-bounded structures trivially have Abelian colours, since the automorphism group of every colour class is either trivial or $\mathbb{Z}_2$. Hence CPT capture polynomial-time on 2-bounded structures. Further, since also multipedes are 2-bounded structures, this resolves the above-mentioned problem posed by Blass, Gurevich, and Shelah (cf. [7, Question 5.12, p. 1115]).

Corollary 11. *The isomorphism problem for multipedes is* CPT*-definable.*

5 Symmetric Circuits

Any property of finite τ-structures can be considered as a sequence of Boolean functions $(f_n)_{n\in\mathbb{N}}$ where f_n takes as inputs the truth values of the atomic τ-formulae on a given structure $\mathfrak{A}$ with universe $[n] = \{0, \dots, n-1\}$, and returns either 0 or 1, depending on whether or not $\mathfrak{A}$ satisfies the given property. To represent really a property of structures of size n, and not of ordered presentations of these, the function f_n must be invariant under any permutation of the universe $[n]$.

Clearly every property of finite structures that is decidable in polynomial time is also decidable by a p-uniform sequence $(C_n)_{n\in\mathbb{N}}$ of polynomial-size Boolean circuits that are invariant in the semantic sense just described. More precisely, every permutation $\pi \in S_n$ of the universe $[n]$ induces a permutation of the input gates of C_n, and the value computed by the circuit C_n does not change if the values $\overline{a}$ of the input gates (representing a structure $\mathfrak{A}$ with universe $[n]$) are changed to $\pi\overline{a}$ (representing the structure $\pi\mathfrak{A} \cong \mathfrak{A}$). Such circuits, and circuit families, are called *invariant*.

On the other side, if we translate a formula from a simple logical language, say, first-order logic or fixed-point logic, into a sequence of circuits, such that circuit C_n simulates the evaluation of the formula on input structures with universe $[n]$, then these circuits are of course invariant, of polynomial size, and in fact p-uniform in the sense that the circuit C_n is computable in polynomial time in n. But moreover such circuits satisfy the stronger property that every permutation of the input universe induces in fact an *automorphism* of the circuit. Circuits with this property are called *symmetric*. Obviously, symmetric circuits are invariant, and it is easy to see that the converse is not true. However, it is *a priori* not clear whether polynomial size symmetric circuits (over a given basis) define a weaker computation model than invariant ones.

For logics with counting, such as FPC, it is natural to consider circuits with threshold or majority gates. Notice that the extension of the standard Boolean basis by majority gates does not change the power of polynomial-size circuit

families, but it can make a difference for specific classes of circuits, such as bounded-depth circuits or symmetric ones. It is a simple observation that every sentence of FPC is equivalent to a p-uniform sequence of symmetric circuits with majority gates.

Can also Choiceless Polynomial Time be translated into such circuits families? If this were the case, then one might use methods from circuit complexity theory to study the power of CPT and understanding its connection with PTIME. With this question in mind, Anderson and Dawar [2] set out to study the power of polynomial-size families of symmetric circuits, both over the standard Boolean basis, and the extension by majority or threshold gates. However, their main results show that symmetric circuits (of polynomial size) are too weak for CPT. In fact, in the version with threshold gates, they are equivalent to FPC and thus cannot define, say, the CFI-query.

Theorem 12. *A class of finite structures is decided by a p-uniform sequence of symmetric threshold circuits if, and only if, it is definable in fixed-point logic with counting. Similarly, a class of finite structures* $\mathfrak{A}$ *is decided by a p-uniform sequence of Boolean circuits if, and only if, it is definable in least fixed-point logic over their two-sorted expansions* $\mathfrak{A}^* = \mathfrak{A} \cup \langle n, < \rangle$.

This theorem has interesting consequences for the study of Choiceless Polynomial Time. It shows that a translation of CPT programs into equivalent sequences of symmetric threshold circuits cannot be done in a p-uniform way. To put it differently, any p-uniform translation from CPT into equivalent sequences of threshold circuits has to break symmetry in some way.

6 Interpretation Logic

While the common presentations of Choiceless Polynomial Time via the manipulation of hereditarily finite sets, by abstract state machines or terms in BGS-logic is convenient and powerful for the design of abstract computations on structures, it makes the analysis of the expressive power of CPT rather difficult. Standard techniques for the analysis of logical systems as used in finite model theory, for instance those based on Ehrenfeucht-Fraïssé methods, are not directly available. In particular, applications of comprehension terms increase the rank of objects and are difficult to handle by the common logical tools, which are usually restricted to 'flat' objects.

However, as recently shown in [16], one can provide alternative presentations of CPT (with and without counting) that are based on classical model-theoretic techniques. In particular, the 'fancy data structures' of the hereditarily finite sets and the manipulation of comprehension terms can be replaced by traditional first-order interpretations. In this context, counting can then be handled by Härtig quantifiers which are classical quantifiers for cardinality comparison. Thus choiceless computations on finite structures can be captured by *iterations of interpretations*. A run is a sequence of states, each of which is now a *finite* structure of a fixed vocabulary. There is an initial interpretation that produces

the initial state as a structure interpreted in the input structure, and a second interpretation $\mathcal{I}_{\text{step}}$ that always maps the current state to its successor state. Since interpretations need not be one-dimensional they can increase the size of the states. Although one application of an interpretation increases the size only polynomially, without imposing restrictions, the iterated application through a polynomial number of steps could produce states of exponential size.

Polynomial-Time Interpretation Logic, denoted PIL, is obtained by imposing polynomial bounds on the length of such computations and the size of the states. It turns out that PIL has precisely the same expressive power as Choiceless Polynomial Time.

Theorem 13. PIL $\equiv$ CPT.

The equivalence survives also in the absence of counting: Polynomial-Time Interpretation Logic without the Härtig quantifier PIL$^-$ is equivalent to CPT$^-$.

Further, the presentation of CPT in terms of first-order interpretations leads to natural fragments and stratifications of this logic along familiar syntactic parameters. For instance, one can consider the natural restrictions of PIL to k-dimensional interpretations, and/or to interpretations where the domain or equivalence formulae are trivial. It turns out that the iteration of one-dimensional interpretations is in fact equivalent to the familiar relational iteration appearing in the partial fixed-point logic PFP, or equivalently, in the database language while. Thus, without the Härtig quantifier, one-dimensional Polynomial-Time Interpretation Logic turns out to be equivalent to the polynomial-time restriction of PFP, which by means of a classical result due to Abiteboul and Vianu implies that one-dimensional PIL$^-$ is equivalent to LFP if, and only if, PTIME = PSPACE. On the other hand, it is known (see e.g. [26]) that the polynomial-time restriction of PFP with counting is actually equivalent to FPC. It follows that one-dimensional PIL, when evaluated on the expansions of finite structures by an ordered numerical sort, has precisely the expressive power of FPC. One can thus view FPC as a one-dimensional fragment of PIL and CPT. This confirms the intuition that the additional power of Choiceless Polynomial Time over FPC comes from the generalization of relational iteration in a fixed arity (as in fixed-point logics) to iterations of relations of changing arities. Already two-dimensional interpretations give us this additional flexibility of relational iteration and, indeed, two-dimensional PIL turns out to be equivalent to full PIL.

Another interesting aspect is the representation of equality by congruence relations and the passage to quotient structures. One may ask whether these are really necessary for obtaining the full expressive power of PIL. The answer is yes.

In the absence of counting, PIL$^-$ without congruences is equivalent to a previously studied extension of the database language while, called while$_{\text{new}}$ $|_{\text{PTIME}}$ which is known to be strictly weaker than CPT. In the presence of counting, the situation is even more intriguing. On any class of structures of bounded colour class size, PIL without congruences can be simulated by CPT-programs

that access only hereditarily finite sets of bounded rank. In particular this holds for the class of CFI-graphs. Since Dawar, Richerby, and Rossman prove in [14] that the CFI-query is definable in CPT, but not by programs of bounded rank, this separates also congruence-free PIL from full PIL. Hence with or without counting, congruences are really essential for reaching the full power of PIL.

7 Challenges for Future Research

Of course the main open problem concerning Choiceless Polynomial Time remains the question whether it captures all of PTIME. But no matter whether or not this is the case, there are a number of interesting open problems that seem to be within the reach of current techniques.

7.1 A Characterization Without Explicit Polynomial Bounds

An unsatisfactory point in the definition of both CPT and PIL is the requirement for explicit polynomial bounds on the running time, the number of active elements or the size of the interpreted structures appearing in a run. It would be desirable to have a characterization of CPT that does not depend on such explicit bounds, but guarantees polynomial-time evaluation implicitly, by construction, as in classical logical approaches such as fixed-point logics.

7.2 Polynomial-Time Properties of Small Definable Subgraphs

Recall Theorem 8, intuitively stating that CPT expresses all polynomial time properties of definable substructures of logarithmic size. It is quite possible that this theorem can further be strengthened, say, to definable substructures of polylogarithmic size.

Is there a function f with $f(n) = \omega(\log n)$ such that the theorem can be strengthened to hold for all $\ell \leq f(n)$? Or is this even the case for all f with polylogarithmic growth rate?

7.3 Isomorphism of CFI-Graphs and Graphs of Bounded Colour Class Size

We have already mentioned the question of whether isomorphism of graphs of bounded colour class size is in CPT; this is basically only known for graphs with Abelian colours (see Section 4). The CFI-graphs are usually presented as a special case. These are graphs of colour class size four (or more general structures of colour class size two) with Abelian colours, provided that the input graphs are coloured in a certain way. This is not inherent in the construction, but just a convenience of the presentation.

It is an open problem whether the isomorphism of uncoloured CFI-graphs (which is still in polynomial time) is in CPT. Actually, this might be a candidate for separating CPT from PTIME.

7.4 Constraint Satisfaction Problems

Constraint satisfaction problems (CSPs), defined in terms of their constraint language, form a rich family of problems in NP that contains many practically important problems, but is still relatively well behaved and excludes pathological examples of problems such as Ladner's [24] NP-problems that are neither in PTIME nor NP-complete. Indeed, Feder and Vardi's [15] well known Dichotomy Conjecture states that all CSPs are either in PTIME or NP-complete. Bulatov, Jeavons, and Krokhin [9] made a refined conjecture characterising the CSPs in PTIME algebraically.

We may ask which CSPs are *solvable* in CPT. If this class coincides with the class of CSPs conjectured to be PTIME-solvable by Bulatov, Jeavons, and Krokhin, this could be seen as evidence that CPT captures polynomial time, or at least does so on the class of all CSPs.

Atserias, Bulatov, and Dawar [3] characterised the CSPs solvable in FPC as precisely those with a property called *bounded width*. It is known that CPT can solve CSPs of unbounded width; cyclic equation systems are examples. These equations systems are CSPs that belong to a polynomial time solvable class of CSPs known as CSPs with *Mal'tsev polymorphisms* [8]. As a first step towards the general question, we ask whether all CSPs with Mal'tsev constraints are solvable in CPT. We remark that an affirmative answer to this question would imply that isomorphism of graphs of bounded colour class size is in CPT (via a reduction described in [4]).

7.5 A Notion of Symmetric Circuits for CPT

We mentioned in Section 5 the result by Anderson and Dawar that p-uniform symmetric circuits are too weak for CFP, and that therefore any p-uniform translation of CPT into equivalent sequences of threshold circuits has to break symmetry in some way. It is an interesting challenge to see how, and to come up with a circuit model capturing CPT. Anderson and Dawar suggest to consider weaker notions of symmetry, requiring induced automorphisms of the circuit only for certain subgroups of the symmetric group on the input universe.

7.6 Choiceless Polynomial Time versus Rank Logic

Besides Choiceless Polynomial Time, one may consider rank logic to be the most prominent candidate for a logic for PTIME. The relationship between these two logics is, at this point unclear.

Theorem 9 about the solvability of cyclic linear equation systems might provide a handle to separate the two logics. Indeed, while the solvability of (arbitrary) linear equation systems over finite fields can clearly be expressed in rank logic, we see no way how rank logic would be able to deal with solvability problems for CES over rings $\mathbb{Z}_{p^k}$ for $k > 1$. Indeed we conjecture that the solvability of CES over $\mathbb{Z}_4$ might be problem that is definable in CPT but not in rank logic.

References

1. F. Abu Zaid, E. Grädel, M. Grohe, and W. Pakusa. Choiceless Polynomial Time on structures with small Abelian colour classes. In *MFCS 2014*, volume 8634 of *Lecture Notes in Computer Science*, pages 50–62. Springer, 2014.
2. M. Anderson and A. Dawar. On symmetric circuits and fixed-point logics. In *STACS*, pages 41–52, 2014.
3. A. Atserias, A. Bulatov, and A. Dawar. Affine systems of equations and counting infinitary logic. *Theoretical Computer Science*, 410:1666–1683, 2009.
4. C. Berkholz and M. Grohe. Limitations of algebraic approaches to graph isomorphism testing. *ArXiv*, arXiv:1502.05912 [cs.CC], 2015.
5. A. Blass and Y. Gurevich. Strong extension axioms and shelah's zero-one law for choiceless polynomial time. *J. of Symbolic Logic*, 68(1):65–131, 2003.
6. A. Blass, Y. Gurevich, and S. Shelah. Choiceless polynomial time. *Annals of Pure and Applied Logic*, 100:141–187, 1999.
7. A. Blass, Y. Gurevich, and S. Shelah. On polynomial time computation over undered structures. *Journal of Symbolic Logic*, 67:1093–1125, 2002.
8. A. Bulatov and V. Dalmau. A simple algorithm for Mal'tsev constraints. *SIAM Journal on Computing*, 36(1):16–27, 2006.
9. A. Bulatov, P. Jeavons, and A. Krokhin. Classifying the complexity of constraints using finite algebras. *SIAM Journal on Computing*, 34:720–742, 2005.
10. J. Cai, M. Fürer, and N. Immerman. An optimal lower bound on the number of variables for graph identification. *Combinatorica*, 12:389–410, 1992.
11. D.G. Corneil and M.K. Goldberg. A non-factorial algorithm for canonical numbering of a graph. *Journal of Algorithms*, 5(3):345–362, 1984.
12. A. Dawar. The nature and power of fixed-point logic with counting. *ACM SIGLOG News*, 2(1):8–21, 2015.
13. A. Dawar, M. Grohe, B. Holm, and B. Laubner. Logics with rank operators. In *Proc. 24th IEEE Symp. on Logic in Computer Science (LICS 09)*, pages 113–122, 2009.
14. A. Dawar, D. Richerby, and B. Rossman. Choiceless Polynomial Time, counting and the Cai-Fürer-Immerman graphs. *Annals of Pure and Applied Logic*, 152:31–50, 2009.
15. T. Féder and M.Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through datalog and group theory. *SIAM Journal on Computing*, 28:57–104, 1998.
16. E. Grädel, Ł. Kaiser, W. Pakusa, and S. Schalthöfer. Characterising Choiceless Polynomial Time with first-order interpretations. In *LICS*, 2015.
17. E. Grädel and M. Otto. Inductive definability with counting on finite structures. In *Computer Science Logic, CSL 92*, volume 702 of *LNCS*, pages 231–247. Springer-Verlag, 1992.
18. E. Grädel and W. Pakusa. Rank logic is dead, long live rank logic! http://arxiv.org/abs/1503.05423, 2015.
19. E. Grädel et al. *Finite Model Theory and Its Applications*. Springer-Verlag, 2007.
20. M. Grohe. Fixed-point definability and polynomial time on graph with excluded minors. *J. ACM*, 59(5):27:1–27:64, 2012.
21. Y. Gurevich and S. Shelah. On finite rigid structures. *J. of Symbolic Logic*, 61:549–562, 1996.
22. B. Holm. *Descriptive Complexity of Linear Algebra*. PhD thesis, University of Cambridge, 2010.

23. N. Immerman. Expresibility as a complexity measure: results and directions. In *Structure in Complexity Theory*, pages 194–202, 1987.
24. R.E. Ladner. On the structure of polynomial time reducibility. *Journal of the ACM*, 22:155–171, 1975.
25. B. Laubner. *The Structure of Graphs and New Logics for the Characterization of Polynomial Time*. PhD thesis, Humboldt-Universitätt zu Berlin, 2011.
26. M. Otto. *Bounded Variable Logics and Counting*. Springer, 1997.
27. B. Rossman. Choiceless computation and symmetry. In A. Blass, N. Dershowitz, and W. Reisig, editors, *Fields of Logic and Computation: Essays Dedicated to Yuri Gurevich on the Occasion of His 70th Birthday*, volume 6300 of *Lecture Notes in Computer Science*, pages 565–580. Springer-Verlag, 2010.