THE BULLETIN OF SYMBOLIC LOGIC
Volume 4, Number 4, Dec. 1998

FINITE VARIABLE LOGICS IN DESCRIPTIVE COMPLEXITY THEORY

MARTIN GROHE

Throughout the development of finite model theory, the fragments of first-order logic with only finitely many variables have played a central role. This survey gives an introduction to the theory of finite variable logics and reports on recent progress in the area.

For each $k \geq 1$ we let L^k be the fragment of first-order logic consisting of all formulas with at most k (free or bound) variables. The logics L^k are the simplest finite-variable logics. Later, we are going to consider infinitary variants and extensions by so-called counting quantifiers.

Finite variable logics have mostly been studied on finite structures. Like the whole area of finite model theory, they have interesting model theoretic, complexity theoretic, and combinatorial aspects. For finite structures, first-order logic is often too expressive, since each finite structure can be characterized up to isomorphism by a single first-order sentence, and each class of finite structures that is closed under isomorphism can be characterized by a first-order theory. The finite variable fragments seem to be promising candidates with the right balance between expressive power and weakness for a model theory of finite structures. This may have motivated Poizat [67] to collect some basic model theoretic properties of the L^k. Around the same time Immerman [45] showed that important complexity classes such as polynomial time (PTIME) or polynomial space (PSPACE) can be characterized as collections of all classes of (ordered) finite structures definable by uniform sequences of first-order formulas with a fixed number of variables and varying quantifier-depth. Although these early results from descriptive complexity theory have been put in much more elegant forms later using so-called fixed-point logics, the importance of the number of variables as a complexity measure remained.

In 1990, Kolaitis and Vardi [52] proved a 0-1 law for the infinitary finite variable logics. As a corollary, they re-proved a result of Blass, Gurevich, and Kozen [7] that there is a 0-1 law for least-fixed point logic. The fact that makes Kolaitis' and Vardi's paper so remarkable is that it uses finite variable logics as a technical tool to obtain results concerning fixed-point logics, which are

Received July 21, 1998.

1079-8986/98/0404-0001/$6.40

central in descriptive complexity. Once they had realized this possibility, Kolaitis, Vardi, and others (see, for example, [1, 2, 17, 54, 53, 63]) developed finite variable logics into the technical framework of a central part of finite model theory. Notably, Abiteboul and Vianu [2] introduced a machine model reflecting precisely the expressive power of these logics and used it to prove a fundamental result which states that the complexity theoretic question of whether PTIME equals PSPACE is equivalent to the purely logical question of whether least fixed-point logic has the same expressive power as partial fixed-point logic. In his Ph.D. thesis, Dawar [12] gives a coherent picture of the theory at this stage, putting some emphasis on model theoretic aspects.

Immerman [45] proposed to extend the finite-variable logics by counting quantifiers such as "there exist at least m elements x". Counting is of a very low computational complexity. The logics L^k can only count up to k, though. So it seems reasonable to extend them by counting quantifiers. Cai, Fürer, and Immerman [9] related the counting logics to the so-called Lehman-Weisfeiler algorithm, a combinatorial algorithm for graph isomorphism. Using Ehrenfeucht-Fraïssé games for the logic, they could refute a long standing conjecture that this algorithm decides isomorphism at least for graphs of bounded valence. Otto [63, 64] embarked on a systematic study of the finite variable logics with counting and lifted most results for the logics without counting to the counting extensions.

More recently, the 2-variable logic L^2 and its variants such as, for example, 2-variable transitive closure logic, have received much attention. The interest in these logics is due to the fact that they already contain modal logic and related logics such as the modal μ-calculus, which are quite important in computer science, but are still very well-behaved compared to the logics with three or more variables. A distinguishing feature of L^2 compared to all L^k for $k > 2$ is the decidability of its satisfiability problem (due to Mortimer [60]). Starting with a paper of Grädel, Kolaitis, and Vardi [24], the complexity of the satisfiability problem for 2-variable logics has been studied systematically [27, 28, 65]. See [26] for a survey on 2-variable logics.

However, our emphasis is on logics with three or more variables. Central issues are the complexity of equivalence testing and the complexity of computing canonical models for finite variable theories. As we will see, these questions are closely connected to fundamental problems of descriptive complexity theory. They are also related to the problem of determining the computational complexity of graph isomorphism.

Let me close this introduction with some hints how to read this survey: The purpose of Section 2 is to informally present three examples illustrating the importance of the finite variable logics.

The main developments in the theory of finite variable logics are presented in a rather condensed form in Sections 3 and 4. Most proofs are only sketched, if given at all. The reader who worries about the details here is referred to the literature (for example, [12, 19, 64]).

Section 5 is concerned with the question of whether we can compute a model for a given L^k-theory. More precisely, we study the complexity of inverting the L^k-invariants. Furthermore, we discuss the abovementioned canonization problem here. The right definition of a canonization function is somewhat subtle, mainly due to the difficulties that arise from encoding finite structures without an ordering. The reader should always be aware of this encoding problem, which is discussed first in Subsection 1.2.

In Section 6 we develop the theory of finite variable logics with counting. Often we just confine ourselves to pointing out the analogy to the logics without counting, but some new aspects of the counting logics are studied in detail. It may be worthwhile to spend some time on getting familiar with fixed-point logic with counting.

The notion of definable canonization is introduced in Section 7. To give the reader at least an idea of the technical side of the results mentioned there, I have demonstrated the details for the simple example of trees. Actually, this example first appears in Section 6 and is then taken up again in the following section.

In Section 8 we give a short introduction to the basic notions of descriptive complexity theory. In particular, we are going to give a precise formulation of the question about a logic that describes, or captures, polynomial time. The section closes with an application of our earlier results to this question.

In the last section we discuss some open problems and directions of further research.

§1. Preliminaries.

Although most of the basic theory can be developed for arbitrary structures, the interesting results only speak about finite ones. *So we restrict our attention to finite structures with finite vocabularies, unless it is explicitly stated otherwise*. It is also convenient to restrict ourselves to *relational vocabularies*; only occasionally we also admit constant symbols.

We use a standard logical notation, as it can for example be found in [20]. The universe of a structure $\boldsymbol{A}$ is denoted by A, and the interpretation of a relation symbol R in $\boldsymbol{A}$ by $R^{\boldsymbol{A}}$. The expansion of a structure $\boldsymbol{A}$ by a relation R^* is denoted by $(\boldsymbol{A}, R^*)$.

Unless explicitly stated otherwise, τ always denotes an arbitrary vocabulary and k, l positive integers.

We always assume classes of structures to be closed under isomorphism. In the same sense that sentences of a logic define classes of structures, formulas with free variables define *queries*. An *l-ary query of vocabulary τ* is a mapping ρ that associates an l-ary relation $\rho^{\boldsymbol{A}}$ with each τ-structure $\boldsymbol{A}$ in such a way that each isomorphism f between two τ-structures $\boldsymbol{A}$ and $\boldsymbol{B}$ is also an isomorphism between the expanded structures $(\boldsymbol{A}, \rho^{\boldsymbol{A}})$ and $(\boldsymbol{B}, \rho^{\boldsymbol{B}})$. Classes of structures of the same vocabulary, which are closed under isomorphism by our general assumption, are also called *Boolean queries*.

$\mathcal{F}$ denotes the class of all (finite) structures. For each class $\mathcal{C}$ of structures and for each vocabulary τ we let $\mathcal{C}[\tau]$ be the class of all τ-structures in $\mathcal{C}$.

We always let E denote a binary relation symbol. *Graphs* are $\{E\}$-structures where E is symmetric and anti-reflexive. Thus for us graphs are undirected and loop-free without multiple edges.

An *ordered structure* is a structure $\boldsymbol{A}$ whose vocabulary contains the binary relation symbol $\leq$ and for which $\leq^{\boldsymbol{A}}$ is a linear order on A. The relation symbol $\leq$ is reserved for its use in ordered structures. $\mathcal{O}$ denotes the class of all ordered structures.

Tuples are written without brackets or commas, as for example $a_1 \dots a_l$. We use $\bar{a}$ as an abbreviation for a tuple $a_1 \dots a_l$, for some $l \geq 1$. As usual, functions and partial functions are considered as sets of pairs. For intervals of integers, we often use notations like $[1, n]$ $(= \{1, \dots, n\})$ or $[1, n[$ $(= \{1, \dots, n-1\})$.

$\{0, 1\}^*$ denotes the set of all finite words over the alphabet $\{0, 1\}$. The length of a word $\boldsymbol{v} \in \{0, 1\}^*$ is denoted by $|\boldsymbol{v}|$, and the concatenation of two words $\boldsymbol{v}, \boldsymbol{w} \in \{0, 1\}^*$ by $\boldsymbol{vw}$. A *language* over $\{0, 1\}$ is a subset of $\{0, 1\}^*$.

1.1. Complexity classes. I assume that the reader is familiar with the basics of complexity theory and has heard of some of the common complexity classes, such as LOGSPACE (logarithmic space), PTIME (polynomial time), NPTIME (nondeterministic polynomial time), and PSPACE (polynomial space). We consider complexity classes as classes of languages over the alphabet $\{0, 1\}$.

To do computations on objects such as finite structures, formulas, polynomials we need to encode these objects by words over $\{0, 1\}$. An *encoding* for a set A is an injective function $e \colon A \to \{0, 1\}^*$. We always encode natural numbers in unary and extend this to an encoding of integers and sequences of integers.[1] Polynomials with integer coefficients can be identified with certain sequences of integers. Formulas can be considered as words over some richer alphabet and thus easily be translated to words over $\{0, 1\}$.

Encoding structures is more problematic.

1.2. Encoding finite structures. We generalize the common encoding of graphs by adjacency matrices to arbitrary finite structures. We need to be careful, though.

Consider the graph $\boldsymbol{A} = (A, E^{\boldsymbol{A}})$. Once we are given an enumeration of the elements of A, say, by $A = \{a_1, \dots, a_n\}$, we can define an *adjacency matrix* of $\boldsymbol{A}$ to be the $(n \times n)$-matrix whose (i, j)th entry is 1 if $E^{\boldsymbol{A}} a_i a_j$ and 0 otherwise. But since there are $n!$ different ways to enumerate the set A we obtain up to $n!$ different adjacency matrices for $\boldsymbol{A}$ instead of just one as

[1]The unary encoding is needed to obtain the complexity bounds in Theorem 5.7 For the rest of the article, it is inessential

we would like to. We can only resolve this problem if the order in which to enumerate the elements comes along with the graph.

Thus for an ordered graph we can find a canonical adjacency matrix corresponding to the enumeration of A according to the order $\leq^{\boldsymbol{A}}$. Of course we can consider the matrix as a $\{0, 1\}$-word of length $|A|^2$. This, at least, gives us an encoding of ordered graphs, which can easily be generalized to arbitrary *ordered structures* (see [19] for details). We obtain an encoding $\boldsymbol{e}\colon \mathcal{O} \to \{0, 1\}^*$.[2]

With arbitrary structures $\boldsymbol{A}$ we can only associate the language

$$\mathcal{L}(\boldsymbol{A}) = \{\, \boldsymbol{e}((\boldsymbol{A}, \leq^{\boldsymbol{A}})) \mid \leq^{\boldsymbol{A}} \text{ is a linear order of } A \,\}.$$

Algorithms on arbitrary structures $\boldsymbol{A}$ expect a word $\boldsymbol{w} \in \mathcal{L}(\boldsymbol{A})$ as input. Their output should not depend on the particular representation $\boldsymbol{w}$, but only on the isomorphism type of the input structure $\boldsymbol{A}$. In other words, they are supposed to work *isomorphism invariant*. For algorithms producing a yes-no answer, this means that the answer is supposed to be the same for every $\boldsymbol{w} \in \mathcal{L}(\boldsymbol{A})$. If the answer is an element or a relation on the input structure, then when the representation $\boldsymbol{w}$ is changed by some permutation, the output should change by the same permutation.

Of course isomorphism invariance of an algorithm has to be verified, but for most algorithms considered here it is obvious.

1.3. Theories and types. Let L be a logic. We denote the set of all L-formulas of vocabulary τ by $\mathsf{L}[\tau]$.

An $\mathsf{L}[\tau]$-*theory* is a satisfiable set of $\mathsf{L}[\tau]$-sentences that is closed under implication. An $\mathsf{L}[\tau]$-theory T is *complete* if for each sentence $\varphi \in \mathsf{L}[\tau]$ either φ or $\neg\varphi$ belongs to T. The $\mathsf{L}[\tau]$-theory of a τ-structure $\boldsymbol{A}$, denoted by $\mathrm{Th}_{\mathsf{L}}(\boldsymbol{A})$, is the set of all $\mathsf{L}[\tau]$-sentences holding in $\boldsymbol{A}$. Two τ-structures $\boldsymbol{A}$ and $\boldsymbol{B}$ are L-*equivalent* if they have the same $\mathsf{L}[\tau]$-theory. We then write $\boldsymbol{A} \equiv_{\mathsf{L}} \boldsymbol{B}$.

An $\mathsf{L}[\tau]$-l-*type*, for some $l \geq 1$, is a maximal satisfiable set of $\mathsf{L}[\tau]$-formulas whose free variables are contained in some fixed set $\{x_1, \dots, x_l\}$. The $\mathsf{L}[\tau]$-type of an l-tuple $\bar{a}$ in a τ-structure $\boldsymbol{A}$ is the set of all $\mathsf{L}[\tau]$-formulas $\varphi(\bar{x})$ such that $\boldsymbol{A} \models \varphi(\bar{a})$. It is denoted by $\mathrm{tp}_{\mathsf{L}}(\bar{a}, \boldsymbol{A})$ or, if $\boldsymbol{A}$ is clear from the context, by $\mathrm{tp}_{\mathsf{L}}(\bar{a})$. We denote types by $\Theta, \Theta', \dots$ or $\Theta(\bar{x}), \dots$ if we want to indicate which variables are used.

An *atomic l-type of vocabulary* τ is a maximal satisfiable set of atomic and negated atomic formulas of vocabulary τ in some fixed set $\{x_1, \dots, x_l\}$ of variables. The atomic type of a tuple $\bar{a}$ in a τ-structure $\boldsymbol{A}$, denoted by $\mathrm{atp}(\bar{a}, \boldsymbol{A})$ or just $\mathrm{atp}(\bar{a})$, is the set of all atomic and negated atomic formulas $\alpha(\bar{x})$ such that $\boldsymbol{A} \models \alpha(\bar{a})$. Note that, since we only admit finite relational vocabularies, atomic types are always finite.

[2]To make the mapping $\boldsymbol{e}$ injective we include a description of the vocabulary of a structure in its code; usually this is not necessary because one only uses encodings for classes $\mathcal{O}[\tau]$, where τ is fixed.

We usually omit the vocabulary when referring to an $\mathsf{L}[\tau]$-theory or $\mathsf{L}[\tau]$-type and just speak of L-theories and L-types. Likewise, we usually do not explicitly mention the vocabulary of an atomic type.

We say that a tuple $\bar{a}$ in a structure $\boldsymbol{A}$ *realizes* a type or atomic type Θ if for all $\varphi(\bar{x}) \in \Theta$ we have $\boldsymbol{A} \models \varphi(\bar{a})$. A type Θ is *realized* in a structure $\boldsymbol{A}$ if there is a tuple $\bar{a} \in A$ realizing Θ.

1.4. Finite variable logics. Recall from the introduction that L^k is the fragment of first-order logic FO whose formulas contain at most k variables. More formally, we fix a set $\{v_1, \dots, v_k\}$ of variables. Then the set of L^k-formulas is obtained by the following formula formation rules:

$$\frac{}{\varphi} \quad \text{where } \varphi \text{ is an atomic formula with variables among } v_1, \dots, v_k,$$

$$\frac{\varphi}{\neg\varphi}, \quad \frac{\varphi, \psi}{\varphi \vee \psi}, \quad \frac{\varphi}{\exists v_i\ \varphi} \quad \text{where } i \in [1, k].$$

The semantics is the usual first-order semantics. We use the symbols $\wedge$, $\rightarrow$, $\leftrightarrow$, $\forall$ as abbreviations with the usual meaning and symbols like x, y, x_i, y' as abbreviations for any of the variables $v_1, \dots, v_k$.

It is important to note that we may *re-use* variables in L^k-formulas. For example, this enables us to say "there is a path of length 5 between two vertices x, y of a graph" with just 3 variables:

$$\exists z(Exz \wedge \exists x(Ezx \wedge \exists z(Exz \wedge \exists x(Ezx \wedge Exy)))).$$

We are going to see that we cannot say things like "there are at least $k + 1$ elements" in the logic L^k. *Finite variable logics with counting* are designed to overcome this weakness. The set of C^k-formulas is obtained by the following formula formation rules:

$$\frac{}{\varphi} \quad \text{where } \varphi \text{ is an atomic formula with variables among } v_1, \dots, v_k,$$

$$\frac{\varphi}{\neg\varphi}, \quad \frac{\varphi, \psi}{\varphi \vee \psi}, \quad \frac{\varphi}{\exists^{\geq m} v_i\ \varphi} \quad \text{where } i \in [1, k] \text{ and } m \in \mathbb{N}.$$

The mexaning of $\exists^{\geq m} v_i\ \varphi$ is "there are at least m elements v_i such that φ holds". Again we use the usual abbreviations, and in addition variations of the quantifier $\exists^{\leq m}$ such as $\exists^{=m}$ or $\exists^{<m}$.

§2. Finite variable logics in combinatorics and computer science—three examples. This section illustrates three important themes in the theory of finite variable logics.

Reasoning in finite variable logics, in particular in the presence of counting, is typical combinatorial reasoning. Our first example on projective planes shows that certain natural results and questions on finite variable theories are closely related to classical combinatorial questions.

One of the central issues in combinatorics is to derive global statements on finite structures from certain combinatorial conditions, such as "any two lines of a geometry intersect in precisely one vertex". Very often, such combinatorial conditions can be naturally expressed in our finite variable logics. So we may ask in how far finite variable theories determine their models. More precisely, we are interested in the relation between the orbits of the automorphism group of a structure and the types in the finite variable logics it realizes, and the relation between isomorphism of two structures and equivalence of these structures in our logics. In Subsection 2.2 we will see that very often the two sides coincide.

One advantage of describing global structural information in our logics is that this may give us feasible algorithms. For example, equivalence of finite structures in a finite variable logic can be tested in polynomial time (see Theorems 4.12 and 6.18), whereas it is conjectured that there is no polynomial time algorithm testing isomorphism. The third example of this section shows why the number of variables plays an important role when it comes to complexity theoretic considerations.

2.1. Projective planes. A (*finite*) *projective plane* of *order* $n \geq 2$ is a two-sorted structure (P, L, I) consisting of a set P of *points*, a set L of *lines*, and a binary *incidence* relation $I \subseteq P \times L$ such that:

(i) There are $n^2 + n + 1$ points.
(ii) Each line is incident to $n + 1$ points.
(iii) For any two distinct points there is exactly one line that is incident to both of them.

This is not the most common definition of a projective plane, but it immediately shows that for each $n \geq 2$ there is a C^3-sentence π_n whose models are the projective planes of order n. A straightforward computation shows that in a projective plane of order n there are $(n^2 + n + 1)$ lines. It is then easy to derive that any two lines intersect in precisely one point.

Using the game techniques we are going to introduce later it is not hard to prove the following (see Example 6.8):

Proposition 2.1. *Let $n \geq 2$. Then any two projective planes of order n are C^3-equivalent. Thus there is a complete C^3-theory Π_n whose models are the projective planes of order n.*

It is a major open problem to determine for which orders there are projective planes. It is known that there are planes of prime and prime-power order, and that there are no planes of order n where $n \equiv 1 \pmod 4$ or $n \equiv 2 \pmod 4$ and n is not the sum of two squares, but not much more. For example, it is not known whether there exists a projective plane of order 12.

If we had an efficient algorithm to compute a model of a given C^3-sentence, we could apply this algorithm to π_{12} to find out if there is a projective plane of order 12. But of course there is no such algorithm; it is well-known that satisfiability of first-order sentences with three variables is undecidable (see, for example, [8]). However, the situation is different for complete C^3-theories. Given a reasonable finite representation of a complete C^k-theory (see below for such a representation), we can compute a model of this theory. The reason is that such a theory contains information on the size of its models, and we can simply search the space of all models of this size. Unfortunately, this algorithm is highly inefficient. For example, a projective plane of order 12 (model of the theory Π_{12}) contains 157 points and 157 lines, and there are 2^{157^2} possible incidence relations between them.

The question if there are efficient algorithms to compute models for given complete L^k or C^k-theories is one of the main topics of this survey.

As a matter of fact, the proof of Proposition 2.1, as it is given in Example 6.8, yields another interesting statement about the C^3-theory of projective planes: Projective planes are C^3-*regular*, that is, for all 3-tuples $\bar{a}, \bar{b}$ in a projective plane we have

$$\mathrm{atp}(\bar{a}) = \mathrm{atp}(\bar{b}) \Longrightarrow \mathrm{tp}_{C^3}(\bar{a}) = \mathrm{tp}_{C^3}(\bar{b}).$$

REMARK 2.2. It is easy to extend these results on projective planes to all square 2-designs. A 2-(v, k, λ) *design* is an incidence structure (V, B, I) where $|V| = v$, each *block* $X \in B$ is incident with precisely k *points* $x \in V$, and for each pair of distinct points there are precisely λ blocks incident with both of them. It is a *square* design if $|V| = |B|$. For example, projective planes of order n are are square 2-$(n^2 + n + 1, n + 1, 1)$ designs.

It is not hard to prove that for all parameters $v \geq 1$, $1 \leq k < v$, $\lambda \geq 0$ there is a complete C^3-theory whose models are the 2-(v, k, λ) designs. Furthermore, square 2-designs are C^3-regular.

The interesting fact is that the last statement has a converse. Let us a call an incidence structure (V, B, I) *trivial* if $I = \emptyset$ or $I = V \times B$. Then a non-trivial incidence structure is C^3-regular if, and only if, it is a square 2-design.

It can moreover be proved that an incidence structure is C^2-regular if, and only if, it is a 1-Design and that it is C^4-regular if, and only if, it is homogeneous.

Proving these results requires game theoretic arguments, similar to those used in Example 6.8, combined with a few simple facts about designs, as they can, for example, be found in [10].

Similar results can be proved for graphs.

2.2. Isomorphism testing. An important algorithmic problem is to decide whether two given graphs are isomorphic. Rather than testing isomorphism directly, many algorithms try to compute the orbits of the automorphism group of a given graph. This of course suffices for the purpose of isomorphism testing, because two connected graphs are isomorphic if their disjoint union has an orbit that contains vertices from both of them.

The *color refinement algorithm* starts by labeling each vertex of a graph by its valence (that is, the number of its neighbors).

Then, in each iteration step, the labelling obtained after the previous step is refined by extending the label of each node by the multiset of labels of its neighbors.

If the equivalence relation induced by the labelling is not refined in an iteration step, the algorithm stops after this step. On a finite graph, this eventually happens, yielding the so called *stable coloring*. The label a vertex carries in the stable coloring is called the *stable color* of the vertex, and the equivalence classes of vertices of the same stable color are called the *color classes* of the graph.

The connection to finite-variable logics is made through the following proposition:

PROPOSITION 2.3 (Immerman, Lander [49]). *Let $\boldsymbol{A}$, $\boldsymbol{B}$ be graphs and $a \in A$, $b \in B$ vertices. Then a and b have the same C^2-type if, and only if, they have the same stable color.*

Let me only indicate how to prove the forward direction here: Suppose a and b have the same C^2-type, that is, satisfy the same C^2-formulas with one free variable. Then they have the same valence because they satisfy the same formulas of the form $\varphi_0(x) = \exists^{\geq m} y\ Exy$. Furthermore, they get the same label in the first iteration step because they satisfy the same formulas of the form

$$\varphi_1(x) = \exists^{\geq m} y(Exy \wedge \exists^{\geq n} x\ Eyx).$$

Continuing like this, we see that in each iteration step they get the same label. Thus they have the same stable color.

Babai, Erdős, and Selkow [3] have shown that for almost all graphs all color classes have size one, thus for almost all graphs the color refinement algorithm yields a "canonical labelling" of the vertices. We are going to discuss the notion of a canonical labelling in detail later. "Almost all graphs" means that the fraction of graphs of size n having color class size 1 of all graphs of size n tends to 1 with growing n.

Formally, for each class $\mathcal{C}$ of graphs and $n \geq 1$ we let

$$\mu_n(\mathcal{C}) = \frac{|\{\, \boldsymbol{G} \in \mathcal{C} \mid G = [1,n]\,\}|}{|\{\, \boldsymbol{G} \in \mathcal{G} \mid G = [1,n]\,\}|},$$

where $\mathcal{G}$ denotes the class of all graphs. We let $\mu(\mathcal{C}) = \lim_{n\to\infty} \mu_n(\mathcal{C})$ if it exists. Then we have:

THEOREM 2.4 (Babai, Erdös, and Selkow [3]). *There is a class $\mathcal{C}$ of graphs such that $\mu(\mathcal{C}) = 1$ and for all $\boldsymbol{G} \in \mathcal{C}$, all color classes of $\boldsymbol{G}$ have size 1, and the color refinement algorithm reaches the stable coloring after the second refinement step.*

Thus almost all graphs are rigid, and their vertices all have distinct C^2-types. Furthermore, we obtain the following:

COROLLARY 2.5. *On almost all graphs isomorphism coincides with* C^2*-equivalence.*

More precisely, there is a class $\mathcal{C}$ *of graphs with* $\mu(\mathcal{C}) = 1$ *such that for all graphs* $\boldsymbol{A}, \boldsymbol{B} \in \mathcal{C}$ *we have*

$$\boldsymbol{A} \cong \boldsymbol{B} \iff \boldsymbol{A} \equiv_{\mathsf{C}^2} \boldsymbol{B}.$$

REMARK 2.6. There is no analogous result without counting. On the contrary, if $\mathcal{C}$ is a class of graphs such that for all $\boldsymbol{A}, \boldsymbol{B} \in \mathcal{C}$ we have

$$\boldsymbol{A} \cong \boldsymbol{B} \iff \boldsymbol{A} \equiv_{\mathsf{L}^k} \boldsymbol{B}$$

then $\mu(\mathcal{C}) = 0$. This follows easily from results of Kolaitis and Vardi [52].

2.3. Model checking. For any logic L, we let MODEL-CHECKING FOR L be the following problem:

Input: A sentence $\varphi \in \mathsf{L}$ and an $\{E\}$-structure $\boldsymbol{A}$.

Question: Does $\boldsymbol{A}$ satisfy φ?

We could actually define a separate MODEL-CHECKING problem for each vocabulary τ, but this would not give us any further insights.

The MODEL-CHECKING problem shows up in various situations. For example, $\boldsymbol{A}$ may be a database and φ a query to be evaluated in this database. Or $\boldsymbol{A}$ may be the state-space of a program (that is, the set of all states in which the program may be), equipped with a reachability relation, and the sentence φ may specify a certain property the program is supposed to have, say, to stop on every possible execution path. Currently, this so-called "model-checking" approach to program and hardware verification is of great practical importance. Since databases and state-spaces of programs are usually quite big, the complexity issue comes up naturally.

Unfortunately, for first-order logic the complexity of MODEL-CHECKING turns out to be quite high:

THEOREM 2.7 (Vardi [70]).
MODEL-CHECKING FOR FO *is* PSPACE*-complete.*

Example 2.8 below shows why model-checking can be done in PSPACE. To prove hardness one reduces the well-known QUANTIFIED BOOLEAN FORMULA problem (see, for example, [66]) to MODEL-CHECKING FOR FO.

It is important to note that the formula φ is part of the input of the model-checking problem. For a *fixed* first-order formula φ_0, the question of whether a given structure $\boldsymbol{A}$ satisfies φ_0 can be decided in $\mathsf{LOGSPACE}$ and thus has a much lower complexity.

EXAMPLE 2.8. The following first-order sentence says that *any two vertices of an* $\{E\}$*-structure are connected by a path of length* 8:

$$\varphi = \forall x \forall y \exists z_1 \dots \exists z_7 (Exz_1 \wedge Ez_1z_2 \wedge \dots \wedge Ez_7y). \tag{2.1}$$

An obvious way to verify that φ holds in a structure $\boldsymbol{A} = (A, E^{\boldsymbol{A}})$ is to check for all pairs $ab \in A^2$ whether there exists a tuple $c_1 \dots c_7 \in A^7$ such that

$$ac_1, c_1c_2, \dots, c_7b \in E^{\boldsymbol{A}}.$$

Essentially, this amounts to enumerating all tuples $abc_1 \dots c_7 \in A^9$ and check whether the entries in the adjacency matrix of the input structure that correspond to the pairs ac_1, c_1c_2, $\dots$, c_7b are 1. Since only one tuple at a time has to be stored, this only requires $\mathsf{Space}(O(\log(|A|)))$ (storing the current tuple requires $\mathsf{Space}(9\lceil\log(|A|)\rceil)$, and a little more space is needed to do the arithmetic). However, it does require at least $\mathsf{Time}(|A|^9)$.

It is straightforward to generalize this example to obtain an algorithm that requires $\mathsf{Space}(O(m\log(n)))$ and $\mathsf{Time}(O(n^m))$ to check whether a first-order formula of size m holds in a structure of size n.

Nevertheless, we all know that it does not require $\mathsf{Time}(n^9)$ to find out whether all pairs of vertices of an $\{E\}$-structure of size n are connected by a path of length 8. Performing some simple syntactical operations on φ we obtain the equivalent formula

$$\psi = \forall x \forall y \exists z_1 (Exz_1 \wedge \exists z_2 (Ez_1z_2 \wedge \exists z_3 (\dots \exists z_7 (Ez_6z_7 \wedge Ez_7y) \dots))).$$

Note that ψ does not have a subformula containing more than three free variables. This property is the key to a much more efficient algorithm to check whether ψ holds in a structure $\boldsymbol{A}$ than the one we have suggested for φ above. We start with the innermost formula $\psi_1(z_6, z_7, y) = Ez_6z_7 \wedge Ez_7y$ and compute the set S_1 of all triples $c_6c_7b \in A$ satisfying ψ_1. This requires $\mathsf{Time}(O(|A|^3))$. Then we proceed to the next formula $\psi_2(z_6, y) = \exists z_7\psi_1(z_6, z_7, y)$. We only have to step through the set S_1 once to compute a set S_2 of all pairs $c_6b \in A^2$ satisfying ψ_2. In the third step we produce a set S_3 consisting of all triples $c_5c_6b \in A^3$ satisfying $\psi_3(z_5, z_6, y) = Ez_5z_6 \wedge \psi_2(z_6, y)$. Using S_2, we can still do this in $\mathsf{Time}(O(|A|^3))$. We go on like this until we have obtained a set S_{14} consisting of all pairs $ab \in A^2$ satisfying $\psi_{14}(x, y) = \exists z_1\psi_{13}(x, z_1, y)$. It only remains to check that $S_{14} = A^2$. To conclude, this algorithm checks in $\mathsf{Time}(O(|A|^3))$ whether ψ holds in $\boldsymbol{A}$.

Again, it is not hard to generalize this to an algorithm requiring

$$\mathsf{Time}(O(mn^k))$$

to check whether a first-order formula of size m that does not contain a subformula with more than k free variables holds in a structure of size n.

The connection with finite-variable logics is easily made:

PROPOSITION 2.9. *A first-order formula is equivalent to an* L^k*-formula if, and only if, it is equivalent to a formula that does not contain a subformula with more than k free variables* (*for each* $k \geq 1$).

So we have learned that, for each $k \geq 1$, MODEL-CHECKING FOR L^k can be done in polynomial time. As a matter of fact, this is also true for MODEL-CHECKING FOR C^k, because it is not much easier to find out if a structure contains an element with a certain property than to find out whether it contains at least m elements with this property—in the worst case we have to check the property for each element.

THEOREM 2.10 (Vardi [71]). *For each* $k \geq 2$, MODEL-CHECKING FOR L^k *and* MODEL-CHECKING FOR C^k *are* PTIME*-complete.*

To prove hardness, one can easily reduce the MONOTONE CIRCUIT VALUE problem (see, for example, [66]) to the MODEL-CHECKING problems.

§3. The expressive power.

We now start to develop the finite model theory of finite variable logics. We refer the reader to [12] or [19] for a detailed presentation of the basic material of Sections 3 and 4.

3.1. Inflationary fixed-point logic. In descriptive complexity theory one tries to describe complexity classes by means of logics. For this purpose, first-order logic has turned out to be too weak. Simple queries, such as connectivity of graphs, cannot be defined in first-order logic, although their computational complexity is rather low.

Adding fixed-point constructs is a natural way to increase the limited expressive power of first-order logic. Numerous *fixed-point logics* have been considered, but for now we are only going to introduce one, *inflationary fixed-point logic* IFP.

The class of IFP-formulas is obtained by adding the following formula-formation rule to the usual rules to form first-order formulas (allowing second-order variables):

$$\frac{\varphi}{[\mathrm{IFP}_{\bar{x},X}\varphi]\bar{v}}$$ where $\bar{x}$ and $\bar{v}$ are k-tuples of individual variables and X is a k-ary relation variable (for some $k \geq 1$).

The free variables of the formula $[\mathrm{IFP}_{\bar{x},X}\varphi]\bar{v}$ are the variables in $\bar{v}$ and those of φ other than $\bar{x}$ and X.

To define the semantics of IFP, let $\varphi(\bar{x}, X, \bar{y}, \bar{Z})$ be an IFP-formula whose free individual variables are those occurring in the k-tuple $\bar{x}$ and in the tuple $\bar{y}$ and whose free second-order variables are the k-ary X and those occurring in the tuple $\bar{Z}$. Furthermore, let $\boldsymbol{A}$ be a structure, $\bar{b}$ a tuple of elements of A whose length matches the length of $\bar{y}$, and $\bar{C}$ a tuple of relations on A whose length and arities match $\bar{Z}$. We define a sequence $(X_i)_{i\geq 1}$ by letting $X_0 = \emptyset$ and

$$X_{i+1} = X_i \cup \{\, \bar{a} \in A^k \mid \boldsymbol{A} \models \varphi(\bar{a}, X_i, \bar{b}, \bar{C}) \,\}.$$

We let $X_\infty = \bigcup_{i \geq 0} X_i$. Then for $\bar{a} \in A^k$ we let

$$\boldsymbol{A} \models [\mathrm{IFP}_{\bar{x},X}\varphi(\bar{x}, X, \bar{b}, \bar{C})]\bar{a} \iff \bar{a} \in X_\infty.$$

Inflationary fixed-point logic has been introduced to finite model-theory by Gurevich [34], but has been studied in generalized recursion theory under the title *non-monotone inductive definability* long before (see [61]). Gurevich and Shelah [37] proved that it has the same expressive power as the better known *least fixed-point logic*.

Example 3.1. The following IFP-sentence defines the class of connected graphs:

$$\varphi = \forall y \forall x[\mathrm{IFP}_{x,X} x = y \vee \exists y(Xy \wedge Eyx)]x.$$

Note that the variable x is bound by the fixed-point operator and then re-introduced by the universal quantifier outside, whereas the y introduced by the first universal quantifier is not affected by the fixed-point operator.

To see that φ indeed defines connectivity, let $\boldsymbol{A}$ be a graph and $a, b \in A$. An easy induction shows that X_i consists of all vertices of distance at most $i - 1$ from b. Thus $a \in X_\infty$ if, and only if, a is reachable from b. If this holds for all $a, b \in A$ then $\boldsymbol{A}$ is connected.

IFP owes its importance to the fact that on ordered structures it *captures* PTIME. We will discuss this in more detail later. For now let us just state one direction of this result.

Proposition 3.2. *For each* IFP*-formula* φ *there is a* PTIME*-algorithm that checks whether a given structure is a model of* φ.

The proof is a straightforward induction on φ. Note the difference to the Model-Checking problem discussed in the previous section: In Proposition 3.2 the formula φ is fixed, whereas in Model-Checking it is part of the input. Vardi [70] proved that Model-Checking for IFP is $\mathsf{EXPTIME}$-complete.

3.2. Infinitary finite variable logics. There is a very fruitful connection between fixed-point logics and finite variable logics. It is best to make this connection through infinitary finite variable logics,[3] which have been introduced by Barwise [6]. Kolaitis and Vardi [52] were the first to realize their importance for finite model theory.

As usual, $\mathsf{L}_{\infty\omega}$ denotes the infinitary first-order logic allowing conjunctions and disjunctions over arbitrary sets. $\mathsf{L}^k_{\infty\omega}$ is the fragment of $\mathsf{L}_{\infty\omega}$ consisting of all formulas with at most k variables. Furthermore, $\mathsf{L}^\omega_{\infty\omega} = \bigcup_{k \geq 1} \mathsf{L}^k_{\infty\omega}$.

[3] We will see later (in Corollary 4.6) that we can replace infinitary formulas by theories of finitary formulas.

Note that $\mathsf{L}^{\omega}_{\infty\omega} \neq \mathsf{L}_{\infty\omega}$. Actually, we have the following picture:

$$\begin{array}{ccccccccccc} \mathsf{L}^1 & \subset & \mathsf{L}^2 & \subset & \mathsf{L}^3 & \subset \cdots \subset & \mathsf{FO} & & \\ \| & & \cap & & \cap & & \cap & & \\ \mathsf{L}^1_{\infty\omega} & \subset & \mathsf{L}^2_{\infty\omega} & \subset & \mathsf{L}^3_{\infty\omega} & \subset \cdots \subset & \mathsf{L}^{\omega}_{\infty\omega} & \subset & \mathsf{L}_{\infty\omega}. \end{array} \tag{3.1}$$

Here the symbol $\subset$ is to be read as a strict inclusion in the semantical sense. Strictness will become clear from the examples of the following subsection.

EXAMPLE 3.3. There is an $\mathsf{L}^3_{\infty\omega}$-sentence ψ defining the class of connected graphs. To see this, for each $i \geq 0$ we define an L^3-formula $\varphi_i(x, y)$ saying that there is a path of length i from x to y. We let $\varphi_0(x, y) = (x = y)$ and for $i \geq 0$

$$\varphi_{i+1}(x, y) = \exists z(Exz \wedge \varphi_i(z, y)),$$

where $\varphi_i(z, y)$ is the result of simultaneously replacing x by z and z by x in $\varphi_i(x, y)$ (note that we replace all occurrences, free and bound). Then we let

$$\varphi = \forall x \forall y \bigvee_{i \geq 1} \varphi_i(x, y).$$

Actually, this can be generalized to the following result.

PROPOSITION 3.4 (Kolaitis, Vardi [52]). *For each* IFP*-formula* φ *there exists a* $k \geq 1$ *and an* $\mathsf{L}^k_{\infty\omega}$*-formula* ψ *equivalent to* φ. *In other words,* $\mathsf{IFP} \subseteq \mathsf{L}^{\omega}_{\infty\omega}$.

To prove this, one defines the *stages* of a fixed-point process, that is, the sets X_i, by finite variable first-order formulas and then takes their disjunction.

EXAMPLE 3.5 (Poizat [67]). Each class of ordered $\{E, \leq\}$-structures that is closed under isomorphism is definable by a single $\mathsf{L}^2_{\infty\omega}$-sentence.

We first define an $\mathsf{L}^2_{\infty\omega}$-sentence φ saying that $\leq$ is a linear order.[4] We let

$$\psi = \forall x \forall y \Big((x \leq y \vee y \leq x) \wedge \big((x \leq y \wedge y \leq x) \rightarrow x = y \big) \Big).$$

Note that in a (finite) model $(A, \leq^{\boldsymbol{A}})$ of ψ either $\leq^{\boldsymbol{A}}$ is a linear order or it has a cycle. Furthermore, it has a cycle if, and only if, it contains decreasing sequences

$$a_1 > a_2 > \cdots > a_m$$

of arbitrary length. This can be said with two variables: We let $\varphi_1(x) = (x = x)$ and, for $m \geq 1$,

$$\varphi_{m+1}(x) = \exists y(x \geq y \wedge x \neq y \wedge \varphi_m(y)),$$

[4]Using game techniques similar to those introduced in the next subsection, it is not hard to prove that there is no L^2-sentence saying that $\leq$ is a linear order.

where $\varphi_m(y)$ is the result of simultaneously replacing x by y and y by x in $\varphi_m(x)$. Then for each $m \geq 1$, $\exists x\ \varphi_m(x)$ says that there exists a decreasing sequence of length m. We let

$$\chi = \bigwedge_{m \geq 1} \exists x\ \varphi_m(x).$$

Then $\varphi = \psi \wedge \neg\chi$ says that $\leq$ is a linear order.

Note that in a linear order, the formula $\varphi_m(x)$, for an $m \geq 1$, says that x is at least the mth element. Then

$$\varphi_{=m}(x) = \varphi_m(x) \wedge \neg\varphi_{m+1}(x)$$

says that x is the mth element.

Now for each ordered $\{E, \leq\}$-structure $\boldsymbol{A}$ we can define an $\mathsf{L}^2_{\infty\omega}$-sentence $\varphi_{\boldsymbol{A}}$ that characterizes $\boldsymbol{A}$ up to isomorphism. For example, to say that there is an edge from the sixth to the fourth element we write

$$\exists x \exists y (Exy \wedge \varphi_{=6}(x) \wedge \varphi_{=4}(y)).$$

Finally, we can define a class $\mathcal{C}$ of ordered $\{E, \leq\}$-structures by the $\mathsf{L}^2_{\infty\omega}$-sentence $\bigvee\{\, \varphi_{\boldsymbol{A}} \mid \boldsymbol{A} \in \mathcal{C}\,\}$.

This example shows that there are non-recursive classes of structures definable in $\mathsf{L}^2_{\infty\omega}$. By Proposition 3.2, all classes definable in IFP are PTIME-computable and thus recursive. Thus there are $\mathsf{L}^2_{\infty\omega}$-formulas not equivalent to any IFP-formula, and the inclusion $\mathsf{IFP} \subset \mathsf{L}^{\omega}_{\infty\omega}$ is strict.

In a sense, on ordered structures the infinitary finite-variable logics are too expressive to allow interesting results.

3.3. An Ehrenfeucht-Fraïssé game. Equivalence in the finite variable logics can be characterized by the so-called *pebble game*. This characterization can be used to obtain non-expressibility results, but it is also the core of a number of model theoretic results for the finite variable logics.

DEFINITION 3.6. Let $k \geq 1$ and τ a vocabulary. The *k-pebble game* is played by two players, a *spoiler* and a *duplicator*, on a pair $\boldsymbol{AB}$ of τ-structures.

A *position* of the game is a subset of $A \times B$ of size at most k. The game starts in an initial position and consists of a sequence of rounds.

In each round of the game the spoiler first removes a pair from the current position if its size is k. Let P be the resulting position (of size at most $(k-1)$). Now the spoiler has the choice to make an $\exists$-move or a $\forall$-move. In an *$\exists$-move* he selects an $a \in A$. The duplicator answers by selecting a $b \in B$, and the new position is $P \cup \{ab\}$. In a *$\forall$-move* the spoiler selects a $b \in B$, the duplicator answers by selecting an $a \in A$, and again the new position is $P \cup \{ab\}$.

The duplicator *wins* the game if each position P that occurs is a partial isomorphism between $\boldsymbol{A}$ and $\boldsymbol{B}$ (that is, an isomorphism whose domain is the substructure of $\boldsymbol{A}$ with universe $\{\, a \in A \mid \exists b \in B : ab \in P\,\}$).

If $\boldsymbol{A} = \boldsymbol{B}$, we refer to the game on $\boldsymbol{AB}$ as the game on $\boldsymbol{A}$.

THEOREM 3.7 (Barwise [6], Immerman [45]). *Let $k \geq 1$ and τ a vocabulary. Furthermore, let $\boldsymbol{A}$, $\boldsymbol{B}$ be τ-structures and $\bar{a} = a_1 \dots a_l \in A^l$, $\bar{b} = b_1 \dots b_l \in B^l$ l-tuples, for an $l \in [1, k]$. Then we have*:

(i) $\boldsymbol{A} \equiv_{\mathsf{L}^k_{\infty\omega}} \boldsymbol{B}$ *if, and only if, the duplicator has a winning strategy for the k-pebble game on $\boldsymbol{AB}$ with initial position $\emptyset$.*

(ii) $\mathrm{tp}_{\mathsf{L}^k_{\infty\omega}}(\bar{a}) = \mathrm{tp}_{\mathsf{L}^k_{\infty\omega}}(\bar{b})$ *if, and only if, the duplicator has a winning strategy for the k-pebble game on $\boldsymbol{AB}$ with initial position $\{a_1b_1, \dots, a_lb_l\}$.*

We sketch some simple applications of this result:

EXAMPLE 3.8. There is no $\mathsf{L}^k_{\infty\omega}$-sentence that defines the class of all sets of size k. (Here sets are considered as structures of empty vocabulary.)

For let A be a set of size k and B a set of size $k + 1$. We shall prove that the duplicator has a winning strategy for the k-pebble game on the $\emptyset$-structures (A) and (B). For an $l \leq k$, let $P = \{a_1b_1, \dots, a_lb_l\}$ be a partial isomorphism between (A) and (B), that is, just a partial bijection between the sets. We consider the game in position P. If we can prove that the duplicator can answer each move of the spoiler in such a way that the new position is still a partial bijection, we are done, because this shows that, starting from the empty position, the duplicator can go on and play forever without losing.

If $l = k$ the spoiler first removes a pair from P, but clearly P remains a partial bijection. Thus without loss of generality we can assume that $l < k$. Suppose the spoiler decides to make an $\exists$-move and selects $a \in A$. If $a = a_i$ for some $i \leq l$ then the duplicator answers by selecting b_i, and the new position $\{a_1b_1, \dots, a_lb_l, a_ib_i\} = \{a_1b_1, \dots, a_lb_l\}$ is still a partial bijection. Otherwise, the duplicator selects an arbitrary $b \in B \setminus \{b_1, \dots, b_l\}$, again with the effect that the new position $\{a_1b_1, \dots, a_lb_l, ab\}$ is a partial bijection. Since $|B| \geq k > l$, such a b exists. If the spoiler decides to make an $\forall$-move, the duplicator can proceed similarly, using $|A| \geq k$.

EXAMPLE 3.9. It follows immediately from Example 3.8 that there is no $\mathsf{L}^\omega_{\infty\omega}$-sentence that defines the class EVEN of all sets of even size.

The reader may now use Examples 3.5, 3.8 and 3.9 to verify that all inclusions in (3.1) are strict.

EXAMPLE 3.10. The class of connected graphs is not definable by an $\mathsf{L}^2_{\infty\omega}$-sentence. Let $\boldsymbol{A}$ be the graph consisting of a single cycle of length 4 and $\boldsymbol{B}$ the graph consisting of two disjoint cycles of length 3. $\boldsymbol{A}$ is connected, but $\boldsymbol{B}$ is not. Nevertheless, the duplicator has a winning strategy for the 2-pebble game on $\boldsymbol{AB}$, thus by Theorem 3.7 the two structures satisfy the same $\mathsf{L}^2_{\infty\omega}$-sentences.

We leave it to the reader to find a strategy for the duplicator.

In particular, the last example shows that the k in Proposition 3.4 cannot be chosen to be the number of individual variables of the IFP-formula φ, since Example 3.1 shows that the class of connected graphs can be defined by an IFP-formula with just 2 individual variables. See [54] for a discussion of the finite variable fragments of fixed-point logics.

§4. **The basic theory.** This section is quite technical, but in the end we will have some nice results. The material presented in Subsections 4.1–4.3 is due to Dawar, Lindell, and Weinstein [17]. Independently, Otto [63] has obtained essentially the same results, though he presented them in a different form. Both Dawar et al.'s and Otto's results are based on work of Abiteboul and Vianu [2] and Kolaitis and Vardi [51].

4.1. Defining the winning positions (I). Let $\boldsymbol{A}$ be a finite structure. When does the spoiler win the k-pebble game on $\boldsymbol{A}$ in a position $\{a_1b_1, \dots, a_kb_k\}$?

The spoiler wins the game immediately, or in 0 rounds, if the tuples $\bar{a}$ and $\bar{b}$ have different atomic types. Unraveling the definition of the game, we see that the spoiler can win the game with initial position $\{a_1b_1, \dots, a_kb_k\}$ in at most r rounds if he can already win in at most $(r-1)$ rounds or there is an $i \in [1, k]$ such that:

- there is an $a \in A$ such that for all $b \in A$ he can win the game with initial position $\{a_1b_1, \dots, a_{i-1}b_{i-1}, ab, a_{i+1}b_{i+1}, \dots, a_kb_k\}$ in at most $(r-1)$ rounds,
- or there is a $b \in A$ such that for all $a \in A$ he can win the game with initial position $\{a_1b_1, \dots, a_{i-1}b_{i-1}, ab, a_{i+1}b_{i+1}, \dots, a_kb_k\}$ in at most $(r-1)$ rounds.

This gives us an inductive definition of the sequence $(R_r^{\boldsymbol{A}})_{r\geq 0}$ of $2k$-ary relations on A where $R_r^{\boldsymbol{A}}\bar{a}\bar{b}$ if, and only if, the spoiler can win the game with initial position $\{a_1b_1, \dots, a_kb_k\}$ in at most r rounds. Let $R_\infty^{\boldsymbol{A}} = \bigcup_{r\geq 0} R_r^{\boldsymbol{A}}$. The mapping $R_\infty \colon \boldsymbol{A} \mapsto R_\infty^{\boldsymbol{A}}$ is a $2k$-ary query on the class of τ-structures.

Actually, this query can be defined in the logic IFP. For a k-tuple $\bar{x} = x_1 \dots x_k$ and an $i \in [1, k]$, let $\bar{x}\frac{x}{i}$ denote the tuple $x_1 \dots x_{i-1}xx_{i+1} \dots x_k$. Then

$$\varphi(\bar{x}, \bar{y}) = \Big[\mathrm{IFP}_{\bar{x}\bar{y},X} \neg \bigwedge_{\alpha(\bar{x})\text{ atomic}} (\alpha(\bar{x}) \leftrightarrow \alpha(\bar{y})) \\ \vee \bigvee_{i=1}^{k} \big(\exists x \forall y\, X\bar{x}\tfrac{x}{i}\bar{y}\tfrac{y}{i} \vee \exists y \forall x\, X\bar{x}\tfrac{x}{i}\bar{y}\tfrac{y}{i}\big)\Big]\bar{x}\bar{y}$$

defines R_∞. By Proposition 3.2, this implies that R_∞ can be computed in polynomial time.

Good, but was does the relation $R_\infty^{\boldsymbol{A}}$ really tell us? It holds for two k-tuples $\bar{a}$ and $\bar{b}$ if, and only if, there is a finite number r such that the spoiler can win the game with initial position $\{a_1b_1, \dots, a_kb_k\}$ in at most r rounds.

But of course what we would like to know is whether the spoiler wins at all. Could it not be the case that the spoiler can win the game, but for each r the duplicator has a strategy such that the spoiler needs at least $r+1$-moves to win if she plays according to this strategy?

The answer is no! In the k-pebble game on a structure $\boldsymbol{A}$ of size n there are at most n^{2k} distinct positions. So if the spoiler can reach a certain position of the game from the initial position in some finite number of rounds, he can actually reach the same position in at most n^{2k} rounds. Thus if he can win the game, he can actually win it in at most n^{2k} rounds. This means that the relation $R^{\boldsymbol{A}}_{\infty}$ contains all winning positions for the spoiler.

Of course we can also define the winning positions for the duplicator, and in the following this will be more convenient. Recalling that the duplicator has a winning strategy for the k-pebble game on $\boldsymbol{AB}$ with initial position $\{a_1b_1, \dots, a_kb_k\}$ if, and only if, $\mathrm{tp}_{\mathsf{L}^k_{\infty\omega}}(\bar{a}) = \mathrm{tp}_{\mathsf{L}^k_{\infty\omega}}(\bar{b})$, we thus have:

LEMMA 4.1 (Kolaitis and Vardi [51]). *Let τ be a vocabulary and $k \geq 1$. Let $\sim_k$ be the query that associates the $2k$-ary relation*

$$\{\, \bar{a}\bar{b} \in A^{2k} \mid \mathrm{tp}_{\mathsf{L}^k_{\infty\omega}}(\bar{a}) = \mathrm{tp}_{\mathsf{L}^k_{\infty\omega}}(\bar{b}) \,\}$$

with each τ-structure $\boldsymbol{A}$.

Then $\sim_k$ is definable in IFP *and hence computable in* PTIME.

By Proposition 3.4 the query $\sim_k$ is also definable by an $\mathsf{L}^{\omega}_{\infty\omega}$-formula. As a matter of fact, it is not hard to see that the query can be defined in $\mathsf{L}^{2k}_{\infty\omega}$.

The following Lemma is an improvement of the previous one that will be needed later. A *pre-order* on a set A is a binary relation $\preceq$ on A that is reflexive, transitive, and connex (that is, $a \preceq b$ or $b \preceq a$ for all $a, b \in A$). Associated with each pre-order $\preceq$ on A is the equivalence relation defined by $a \preceq b \wedge b \preceq a$; for brevity we call its equivalence classes the *classes* of $\preceq$.

LEMMA 4.2 (Abiteboul and Vianu [2]). *Let τ be a vocabulary and $k \geq 1$. There is an* IFP*-definable $2k$-ary query $\precsim_k$ that associates with each τ-structure $\boldsymbol{A}$ a pre-order $\precsim^{\boldsymbol{A}}_k$ on A^k whose classes are the equivalence classes of $\sim^{\boldsymbol{A}}_k$.*

Furthermore, $\precsim_k$ orders the $\mathsf{L}^k_{\infty\omega}$-types in the same way in all structures. More precisely, for all $\boldsymbol{A}, \boldsymbol{B} \in \mathcal{F}[\tau]$, $\bar{a}, \bar{a}' \in A^k$, and $\bar{b}, \bar{b}' \in B^k$ such that

$$\mathrm{tp}_{\mathsf{L}^k_{\infty\omega}}(\bar{a}) = \mathrm{tp}_{\mathsf{L}^k_{\infty\omega}}(\bar{b}) \quad \textit{and} \quad \mathrm{tp}_{\mathsf{L}^k_{\infty\omega}}(\bar{a}') = \mathrm{tp}_{\mathsf{L}^k_{\infty\omega}}(\bar{b}')$$

we have

$$\bar{a} \precsim^{\boldsymbol{A}}_k \bar{a}' \iff \bar{b} \precsim^{\boldsymbol{B}}_k \bar{b}'.$$

The proof of this lemma is a variant of the previous one. Whereas $\sim_k$ is obtained by inductively refining the equivalence relation induced by the atomic k-types, $\precsim_k$ is obtained by inductively refining the pre-order induced by some fixed pre-order on the atomic k-types.

4.2. Defining the winning positions (II). In the previous subsection we saw how to define the winning positions for the spoiler and the duplicator in the game on a single structure. Here we prove a slightly different definability result. The proof uses ideas from the classical model theory of infinitary logics that go back to Scott [68] (also see [5]).

LEMMA 4.3 (Dawar, Lindell, Weinstein [17]). *Let $k \geq 1$, τ a vocabulary, $\boldsymbol{A}$ a τ-structure, and $\bar{a} \in A^k$. Then there is an L^k-formula $\varphi^k_{\boldsymbol{A},\bar{a}}(\bar{y})$ such that for all τ-structures $\boldsymbol{B}$ and $\bar{b} \in B^k$ we have*

$$\boldsymbol{B} \models \varphi^k_{\boldsymbol{A},\bar{a}}(\bar{b}) \iff \mathrm{tp}_{\mathsf{L}^k_{\infty\omega}}(\bar{a}, \boldsymbol{A}) = \mathrm{tp}_{\mathsf{L}^k_{\infty\omega}}(\bar{b}, \boldsymbol{B}).$$

PROOF. Recall that the spoiler can win the game on $\boldsymbol{AB}$ with initial position $\{a_1b_1, \dots, a_kb_k\}$ in 0 rounds if $\bar{a}$ and $\bar{b}$ have distinct atomic types. He can win in at most r rounds if he can win in at most $(r-1)$ rounds or if there is an $i \in [1,k]$ such that either there is an $a \in A$ such that for all $b \in B$ he can win the game with initial position $\{a_1b_1, \dots, a_{i-1}b_{i-1}, ab, a_{i+1}b_{i+1}, \dots, a_kb_k\}$ in at most $r-1$ rounds or if there is a $b \in B$ such that for all $a \in A$ he can win the game with initial position $\{a_1b_1, \dots, a_{i-1}b_{i-1}, ab, a_{i+1}b_{i+1}, \dots, a_kb_k\}$ in at most $r-1$ rounds.

Let

$$\psi^{k,0}_{\boldsymbol{A},\bar{a}}(\bar{y}) = \bigvee_{\alpha(\bar{y}) \in \mathrm{atp}(\bar{a})} \neg\alpha(\bar{y})$$

and for all $r \geq 1$

$$\psi^{k,r}_{\boldsymbol{A},\bar{a}}(\bar{y}) = \psi^{k,r-1}_{\boldsymbol{A},\bar{a}}(\bar{y}) \vee \bigvee_{i=1}^{k} \Big(\bigvee_{a \in A} \forall y\ \psi^{k,r-1}_{\boldsymbol{A},\bar{a}\frac{a}{i}}(\bar{y}\tfrac{y}{i}) \vee \exists y \bigwedge_{a \in A} \psi^{k,r-1}_{\boldsymbol{A},\bar{a}\frac{a}{i}}(\bar{y}\tfrac{y}{i}) \Big).$$

Then for all $r \geq 0$, τ-structures $\boldsymbol{B}$ and $\bar{b} \in B^k$ we have:

(4.1) $\boldsymbol{B} \models \psi^{k,r}_{\boldsymbol{A},\bar{a}}(\bar{b}) \iff$ The spoiler can win the k-pebble game on $\boldsymbol{AB}$ with initial position $\{a_1b_1, \dots, a_kb_k\}$ in at most r rounds.

Note that, in particular, this implies $\boldsymbol{A} \models \neg\psi^{k,r}_{\boldsymbol{A},\bar{a}}(\bar{a})$.

Now we would like to argue as in the previous subsection: Since we are dealing with finite structures, the game can only last a certain finite number of rounds. Thus actually there is an r_0 such that $\psi^{k,r_0}_{\boldsymbol{A},\bar{a}}$ suffices. Unfortunately, this argument is not correct, since $\boldsymbol{B}$ may be arbitrarily large.

The correct reasoning is trickier: We first consider the case $\boldsymbol{B} = \boldsymbol{A}$. We know that if the spoiler has a winning strategy for the game on $\boldsymbol{A}$ he can win in at most $r_A = |A|^{2k}$ rounds. Thus

$$\boldsymbol{A} \models \bigwedge_{\bar{a}' \in A^k} \forall \bar{y} (\psi^{k,r_A+1}_{\boldsymbol{A},\bar{a}'}(\bar{y}) \to \psi^{k,r_A}_{\boldsymbol{A},\bar{a}'}(\bar{y})). \tag{4.2}$$

Now let $\boldsymbol{B}$ be an arbitrary structure and $\bar{b} \in B^k$. Suppose that the spoiler has a winning strategy for the k-pebble game on $\boldsymbol{AB}$ with initial position $\{a_1b_1, \dots, a_kb_k\}$.

Case 1. The spoiler can win the game in at most r_A rounds. Then $\boldsymbol{B} \models \psi^{k,r_A}_{\boldsymbol{A},\bar{a}}(\bar{b})$.

Case 2. The spoiler cannot win the game in at most r_A rounds. Then there is a position $P' = \{a'_1b'_1, \dots, a'_kb'_k\}$ of the game such that the spoiler can win the game with initial position P' in $r_A + 1$ rounds, but not in r_A rounds. Thus

$$\boldsymbol{B} \models \psi^{k,r_A+1}_{\boldsymbol{A},\bar{a}'}(\bar{b}') \wedge \neg\psi^{k,r_A}_{\boldsymbol{A},\bar{a}'}(\bar{b}'),$$

implying

$$\boldsymbol{B} \not\models \bigwedge_{\bar{a}' \in A^k} \forall \bar{y}(\psi^{k,r_A+1}_{\boldsymbol{A},\bar{a}'}(\bar{y}) \to \psi^{k,r_A}_{\boldsymbol{A},\bar{a}'}(\bar{y})).$$

Let

$$\varphi^k_{\boldsymbol{A},\bar{a}}(\bar{y}) = \neg\psi^{k,r_A}_{\boldsymbol{A},\bar{a}}(\bar{y}) \wedge \bigwedge_{\bar{a}' \in A^k} \forall \bar{y}(\psi^{k,r_A+1}_{\boldsymbol{A},\bar{a}'}(\bar{y}) \to \psi^{k,r_A}_{\boldsymbol{A},\bar{a}'}(\bar{y}))$$

We have just seen that if the spoiler has a winning strategy for the k-pebble game on $\boldsymbol{AB}$ with initial position $\{a_1b_1, \dots, a_kb_k\}$ we have $\boldsymbol{B} \not\models \varphi^k_{\boldsymbol{A},\bar{a}}(\bar{b})$.

Note on the other hand that by (4.1) and (4.2) we have $\boldsymbol{A} \models \varphi^k_{\boldsymbol{A},\bar{a}}(\bar{a})$. Hence, if the duplicator has a winning strategy for the k-pebble game on $\boldsymbol{AB}$ with initial position $\{a_1b_1, \dots, a_kb_k\}$ then by Theorem 3.7 (ii) we have $\boldsymbol{B} \models \varphi^k_{\boldsymbol{A},\bar{a}}(\bar{b})$. ⊣

Here are some immediate consequences of this result:

Corollary 4.4. *Let $k \geq 1$, τ a vocabulary and $\boldsymbol{A}$ a τ-structure. Then there exists an L^k-sentence $\varphi^k_{\boldsymbol{A}}$ such that for all τ-structures $\boldsymbol{B}$ we have*

$$\boldsymbol{B} \models \varphi^k_{\boldsymbol{A}} \iff \boldsymbol{A} \equiv_{\mathsf{L}^k_{\infty\omega}} \boldsymbol{B}.$$

In particular, the L^k-theory of $\boldsymbol{A}$ is finitely axiomatizable in L^k.

Proof. Let $\varphi^k_{\boldsymbol{A}} = \exists \bar{y} \varphi^k_{\boldsymbol{A},\bar{a}}(\bar{y})$ for an arbitrary k-tuple $\bar{a} \in A^k$. ⊣

Corollary 4.5. *Let $k \geq 1$ and τ a vocabulary. Two τ-structures $\boldsymbol{A}$ and $\boldsymbol{B}$ are $\mathsf{L}^k_{\infty\omega}$-equivalent if, and only if, they are L^k-equivalent.*

Corollary 4.6. *Let $k \geq 1$. Each $\mathsf{L}^k_{\infty\omega}$-sentence is equivalent to an L^k-theory.*

Proof. $\varphi \in \mathsf{L}^k_{\infty\omega}$ is equivalent to $\{ \neg\varphi^k_{\boldsymbol{A}} \in \mathsf{L}^k \mid \boldsymbol{A} \not\models \varphi \}$. ⊣

Analogous results can be formulated for formulas and types instead of sentences and theories. For later reference we state the following:

COROLLARY 4.7. *Let* $\boldsymbol{A}, \boldsymbol{B}$ *be* τ*-structures and* $\bar{a} \in A^l$, $\bar{b} \in B^l$, *where* $l \leq k$. *Then the following statements are equivalent*:

(i) $\mathrm{tp}_{\mathsf{L}^k}(\bar{a}, \boldsymbol{A}) = \mathrm{tp}_{\mathsf{L}^k}(\bar{b}, \boldsymbol{B})$.
(ii) $\mathrm{tp}_{\mathsf{L}^k_{\infty\omega}}(\bar{a}, \boldsymbol{A}) = \mathrm{tp}_{\mathsf{L}^k_{\infty\omega}}(\bar{b}, \boldsymbol{B})$.

4.3. The invariant.

DEFINITION 4.8. A *complete invariant* for an equivalence relation $\equiv$ on a class A is a mapping $\boldsymbol{I}\colon A \to \{0,1\}^*$ such that

$$\forall a, b \in A\colon (a \equiv b \iff \boldsymbol{I}(a) = \boldsymbol{I}(b)).$$

The goal of this subsection is to show that the equivalence relation $\equiv_{\mathsf{L}^k}$ on the class $\mathcal{F}[\tau]$ has a PTIME-computable complete invariant.

Let τ_{I^k} be the vocabulary consisting of binary relation symbols $\leq, E_1, \ldots, E_k$ and a unary relation symbol P_θ for each atomic k-type θ of vocabulary τ.

Recall the definition of the queries $\sim_k$ and $\precsim_k$ from Lemmas 4.1 and 4.2. Since the structure $\boldsymbol{A}$ we refer to will always be clear from the context, in the following we omit the superscript $^{\boldsymbol{A}}$ in $\sim_k^{\boldsymbol{A}}$ and $\precsim_k^{\boldsymbol{A}}$.

Let $\boldsymbol{A}$ be a τ-structure. The $\sim_k$-equivalence class of a k-tuple $\bar{a} \in A^k$ is denoted by $\bar{a}/_{\sim_k}$, and the factorization of A^k by $\sim_k$ by $A^k/_{\sim_k}$. By Corollary 4.7 we have

$$\bar{a} \sim_k^{\boldsymbol{A}} \bar{b} \iff \mathrm{tp}_{\mathsf{L}^k}(\bar{a}) = \mathrm{tp}_{\mathsf{L}^k}(\bar{b}),$$

thus we can view $A^k/_{\sim_k}$ as the set of L^k-k-types realized in $\boldsymbol{A}$.

I^k is the mapping that associates with each τ-structure $\boldsymbol{A}$ the τ_{I^k}-structure $\boldsymbol{I} = I^k(\boldsymbol{A})$ defined as follows:

- $I = A^k/_{\sim_k}$.
- $\leq^{\boldsymbol{I}}$ is the order on I induced by $\precsim_k$, that is,
$$\bar{a}/_{\sim_k} \leq^{\boldsymbol{I}} \bar{b}/_{\sim_k} \iff \bar{a} \precsim_k \bar{b}.$$
- For $i \leq k$, the relation $E_i^{\boldsymbol{I}}$ is defined by
$$E_i^{\boldsymbol{I}}(\bar{a}/_{\sim_k})(\bar{a}'/_{\sim_k}) \iff \exists a' \in A : \bar{a}' \sim_k \bar{a}\tfrac{a'}{i},$$
where $\bar{a}\frac{a'}{i}$ denotes the tuple obtained from $\bar{a}$ by replacing its ith place by a'.
- For each atomic k-type θ, the relation $P_\theta^{\boldsymbol{I}}$ is defined by
$$P_\theta^{\boldsymbol{I}}(\bar{a}/_{\sim_k}) \iff \mathrm{atp}(\bar{a}) = \theta.$$

LEMMA 4.9. *For each* $k \geq 1$ *the mapping* $I^k\colon \mathcal{F} \to \mathcal{O}$ *is well-defined.*

PROOF. Let $\boldsymbol{A} \in \mathcal{F}[\tau]$, $\bar{a}, \bar{b} \in A^k$, and $\boldsymbol{I} = I^k(\boldsymbol{A})$ defined as above.

$\leq^{\boldsymbol{I}}$ is a linear order on $\boldsymbol{I}$ because $\precsim_k$ is a pre-order whose classes are the equivalence classes of $\sim_k$. Thus the image of I^k is contained in $\mathcal{O}$.

If $\bar{a} \sim_k \bar{b}$ then for each $i \in [1,k]$ and for each $a' \in A$ there is a $b' \in A$ such that $\bar{a}\frac{a'}{i} \sim_k \bar{b}\frac{b'}{i}$ (by the game characterization of $\sim_k$). This shows that the relation $E_i^{\boldsymbol{I}}$ is well defined.

Furthermore, since $\sim_k$-equivalent tuples have the same L^k-type they have the same atomic k-type, and thus the relations $P_\theta^{\boldsymbol{I}}$ are well-defined. ⊣

Often, we are only interested in the restriction $I_\tau^k \colon \mathcal{F}[\tau] \to \mathcal{O}[\tau_{I^k}]$ to a particular vocabulary τ.

THEOREM 4.10 (Dawar, Lindell, Weinstein [17]). *Let τ be a vocabulary and $k \geq 1$. Then the following holds*:

(i) *For all τ-structures $\boldsymbol{A}$ and $\boldsymbol{B}$ we have*

$$\boldsymbol{A} \equiv_{\mathsf{L}^k} \boldsymbol{B} \iff \boldsymbol{I}^k(\boldsymbol{A}) \cong \boldsymbol{I}^k(\boldsymbol{B}).$$

(ii) *I_τ^k is* PTIME-*computable.*

PROOF. To prove the forward direction of (i) we assume that $\boldsymbol{A} \equiv_{\mathsf{L}^k} \boldsymbol{B}$. Then, by Corollary 4.5, $\boldsymbol{A}$ and $\boldsymbol{B}$ satisfy the same $\mathsf{L}^k_{\infty\omega}$-sentences. Thus for each L^k-type $\Theta(\bar{x})$ we have

$$\boldsymbol{A} \models \exists \bar{x} \bigwedge \Theta(\bar{x}) \iff \boldsymbol{B} \models \exists \bar{x} \bigwedge \Theta(\bar{x}).$$

Hence $\boldsymbol{A}$ and $\boldsymbol{B}$ realize the same L^k-types, and we can define a bijection f between $A^k/_{\sim_k}$ and $B^k/_{\sim_k}$ that associates equivalence classes corresponding to the same type. By the second part of Lemma 4.2, this bijection is order preserving.

Furthermore, the two structures satisfy the same sentences of the form

$$\forall \bar{x} \Big(\bigwedge \Theta(\bar{x}) \to \exists x' \bigwedge \Theta'(\bar{x}\tfrac{x'}{i}) \Big)$$

for $i \in [1,k]$ and L^k-types $\Theta(\bar{x})$, $\Theta'(\bar{x})$. Thus f is an isomorphism.

For the backward direction, we prove that if $\boldsymbol{I}^k(\boldsymbol{A}) \cong \boldsymbol{I}^k(\boldsymbol{B})$ then the duplicator has a winning strategy for the k-pebble game on $\boldsymbol{AB}$ with initial position $\emptyset$. Let f be an isomorphism between $\boldsymbol{I}^k(\boldsymbol{A})$ and $\boldsymbol{I}^k(\boldsymbol{B})$.

For a position $P = \{a_1b_1, \dots, a_lb_l\}$, where $l \in [1,k]$, of the k-pebble game on $\boldsymbol{AB}$ we let

$$a(P) = a_1 \dots a_l \underbrace{a_l \dots a_l}_{(k-l)\text{ times}} \quad \text{and} \quad b(P) = b_1 \dots b_l \underbrace{b_l \dots b_l}_{(k-l)\text{ times}}.$$

The duplicator's strategy is to play in such a way that for every position $P \neq \emptyset$ occurring in the game we have $f(a(P)/_{\sim_k}) = b(P)/_{\sim_k}$. It is not hard to see that the definition of $\boldsymbol{I}^k$ guarantees that this strategy is successful.

(ii) follows easily from the fact that the query $\precsim_k$ is PTIME-computable. ⊣

COROLLARY 4.11. *Let $k \geq 1$ and τ a vocabulary. Then there is a* PTIME-*computable complete invariant for the equivalence relation $\equiv_{\mathsf{L}^k}$ on $\mathcal{F}[\tau]$.*

PROOF. Recall the definition of the encoding $\boldsymbol{e}$ from page 349. $\boldsymbol{e} \circ I^k$ is the desired complete invariant. ⊣

For convenience, from now on we refer to I^k (rather than $\boldsymbol{e} \circ I^k$) as the L^k-*invariant*. Sometimes we even call $I^k(\boldsymbol{A})$ the L^k-*invariant of* a structure $\boldsymbol{A}$. Note that the invariant gives rise to an encoding of complete L^k-theories. We simply encode a theory T by the invariant of one of its models $\boldsymbol{A}$. We let $I^k(\mathrm{T}) = I^k(\boldsymbol{A})$. Again we may refer to $I^k(\mathrm{T})$ as the *invariant of* T. Our encoding has the nice property that for a given structure $\boldsymbol{A}$ we can compute the code of its L^k-theory in polynomial time.

4.4. The complexity of equivalence. For any logic L we let L-EQUIVALENCE be the following problem:

Input: $\{E\}$-structures $\boldsymbol{A}$ and $\boldsymbol{B}$.
Question: Is $\boldsymbol{A} \equiv_{\mathsf{L}^k} \boldsymbol{B}$?

As for the MODEL CHECKING problem we could define a version of L-EQUIVALENCE for every vocabulary, but we can as well live without it.

As an immediate consequence of Corollary 4.11 we obtain that L^k-EQUIVALENCE is in PTIME. This result is due to Immerman and Lander [49].

THEOREM 4.12 (Grohe [29]). *For all* $k \geq 2$, L^k-EQUIVALENCE *is* PTIME-*complete under quantifier-free reductions.*

Quantifier-free reductions are logical reductions of a very low computational complexity; the result remains true for more familiar types of reductions such as LOGSPACE-reductions or AC_0-reductions.

As the PTIME-completeness of MODEL CHECKING FOR L^k, this result shows the close connection between the finite variable logics and the complexity class PTIME.

There is a different perspective on Theorem 4.12 that underlines its connection with the results of the next section. The theorem implies that *the problem of computing the invariant of a given structure is* PTIME-*complete* (see [66] for a suitable concept of reductions between function problems). To see this, recall that to decide whether two structures are L^k-equivalent we just have to compute their invariants and compare them.

§5. Inversion of the invariants.

The last two sections show, with the exception of Theorem 4.12, the state of the theory, say, in 1993, as it was presented in Dawar's Ph.D. thesis [12]. Of course we have only covered the basic results. In particular, we have omitted the important Abiteboul-Vianu Theorem [2] mentioned in the introduction. For a more comprehensive picture we refer the reader to Dawar's thesis and the books [19, 64].

We now turn to more recent developments.

Recall our discussion in Subsection 2.1: If there are projective planes of order n then they all have the same C^3-theory Π_n. If there are no such planes then we can still write down the "theory" Π_n (it is just no theory because it is unsatisfiable). The fundamental problem is to decide whether Π_n is satisfiable and, if it is, to find models.

Of course the same question occurs for any complete theory. In the following we are going to study it for L^k-theories. C^k-theories will be considered in Subsection 6.5.

We have seen in the last section that complete L^k-theories, which are infinite objects, can be encoded in a natural way by their (finite) invariants. A precise formulation of our question, due to Dawar [12], is:

QUESTION 1. *Is there a recursive inverse* $f : \mathcal{O}[\tau_{I^k}] \to \mathcal{F}[\tau]$ *of* I^k_τ?

Here an *inverse* of a mapping $f : A \to B$ is a mapping $g : B \to A$ such that for all $b \in f(A)$ we have $g(b) \in f^{-1}(b)$.

It is easy to see that Question 1 is equivalent to the question of whether the set $I^k(\mathcal{F}[\tau])$ is recursive.

Purely model theoretic considerations lead to a very similar question. The k-*size* of a structure $\boldsymbol{A}$ is defined to be the number of L^k-k-types realized in $\boldsymbol{A}$, or equivalently, the size of the universe of $I^k(\boldsymbol{A})$. By Theorem 4.10 (i) the k-size of $\boldsymbol{A}$ only depends on $\mathrm{Th}_{\mathsf{L}^k}(\boldsymbol{A})$. Thus we can define the k-size of a complete L^k-theory to be the k-size of its models. It can be seen as a measure for the complexity of the theory.

Dawar [15] observed that there is a function $g : \mathbb{N} \to \mathbb{N}$ such that each complete $\mathsf{L}^k[\tau]$-theory T has a model of size at most $g(k\text{-size}(\mathrm{T}))$. (Remember that an L^k-theory has a finite model by definition.) To see this, note that for all $n \in \mathbb{N}$ there are only finitely many τ_{I^k}-structures of size at most n. Thus there are only finitely many complete L^k-theories of k-size at most n. We can simply let

$$g(n) = \max\{ \min\{ |A| \mid \boldsymbol{A} \in \mathcal{F}[\tau],\ \boldsymbol{A} \models \mathrm{T} \} \mid \mathrm{T} \text{ is a complete } \mathsf{L}^k\text{-theory of } k\text{-size } n \}.$$

This can be seen as a finite downward Löwenheim-Skolem theorem for L^k. However, it is useless if we cannot put any restrictions on the function g. Therefore, Dawar [15] asked:

QUESTION 2. *Is there a recursive function* $g : \mathbb{N} \to \mathbb{N}$ *such that each complete* $\mathsf{L}^k[\tau]$-*theory* T *has a model of size at most* $g(k\text{-}size(\mathrm{T}))$?

It turns out that Question 2 is equivalent to Question 1. For let f be a recursive inverse of I^k_τ. Then g defined by

$$g(n) = \max\{ |f(\boldsymbol{I})| \mid \boldsymbol{I} \in \mathcal{O}[\tau_{I^k}], |I| = n \}$$

satisfies the conditions of Question 2.

On the other hand, let $g : \mathbb{N} \to \mathbb{N}$ satisfy the conditions of Question 2. Let $\preceq$ denote the ordering of finite τ-structures $\boldsymbol{A}$ by the lexicographical ordering of the finite sets $\mathcal{L}(\boldsymbol{A}) \subseteq \{0,1\}^*$ (recall the definition of $\mathcal{L}(\boldsymbol{A})$ from page 349).

We define an inverse f of I^k_τ by letting $f(\boldsymbol{I})$ be the first structure $\boldsymbol{A}$ with respect to $\preceq$ of size at most $g(|I|)$ with $I^k(f(\boldsymbol{I})) = \boldsymbol{I}$ if such a structure exists, and arbitrary otherwise.

Otto [62] proved that for $k = 2$ the answer to Questions 1 and 2 is "yes". Actually, he proved more:

THEOREM 5.1 (Otto [62]). *For each vocabulary τ, there is a* PTIME-*computable inverse for I^2_τ.*

If we are only interested in a recursive rather than a PTIME-computable inverse of I^2_τ, we can proceed as follows, as has been observed by Flum and Ziegler [23] and independently by Otto [64]: Given the invariant $I^2(\boldsymbol{A})$, we can clearly compute the L^2-sentence $\varphi^2_{\boldsymbol{A}}$ that finitely axiomatizes $\mathrm{Th}_{\mathsf{L}^2}(\boldsymbol{A})$ (see Corollary 4.4). By Mortimers Theorem [60], it is decidable whether $\varphi^2_{\boldsymbol{A}}$ has a finite model. This yields a decision procedure for $I^2(\mathcal{F}[\tau])$. The approach via Mortimers theorem does not give us a PTIME-computable inverse, though, and much more work is required to get one.

We can compute the L^k-sentence $\varphi^k_{\boldsymbol{A}}$ from $I^k(\boldsymbol{A})$ for any k. But for $k \geq 3$ the finite satisfiability problem for the logic L^k is no longer decidable, so this does not help.

As far as Question 2 is concerned, Otto's result implies a polynomial upper bound on the size of the minimal model of an L^2-theory in terms of its k-size (actually, Otto gets a linear upper bound). It has been observed by Dawar, Lindell, and Weinstein [17] that for each $k \geq 3$ the L^k-theory of a complete binary tree has a k-size polynomially bounded in the height of the tree, whereas its unique model (the tree itself) is exponentially large in its height. Thus a function g satisfying the requirements of Question 2 (for $k \geq 3$ and $\tau = \{E\}$) must have at least an exponential growth rate.

5.1. Translations. For $k \geq 3$, the answer to Questions 1 and 2 is "no". To see this, we translate Diophantine equations to L^k-theories in such a way that positive integer solutions of the equations translate to models of the theories and vice versa.

The proof of this result consists of a sequence of reductions which are all of a similar form. It is convenient to formulate them in the following abstract setting.

Let us call a triple $\mathrm{A} = (S, M, \models)$ consisting of a class S of "sentences", a class M of "models", and a binary "satisfaction" relation between sentences and models an *abstract logic*. This notion is not intended to be a meaningful definition of what one might reasonably call a logic, but it helps to understand the following results. No confusion will arise from always denoting the satisfaction relation by $\models$. Thus we may just write $\mathrm{A} = (S, M)$. We denote the class of all models of a sentence $s \in S$, that is, the set $\{\, \boldsymbol{m} \in M \mid \boldsymbol{m} \models s \,\}$, by $M(s)$.

Here are some examples of abstract logics we are going to meet:

EXAMPLE 5.2. For each vocabulary τ and $k \geq 1$ we let

$$\mathrm{INV}_k[\tau] = (\mathcal{O}[\tau_{I^k}], \mathcal{F}[\tau])$$

with the satisfaction relation $\boldsymbol{A} \models \boldsymbol{I} \iff I^k(\boldsymbol{A}) = \boldsymbol{I}$.

EXAMPLE 5.3. A *Diophantine equation* is an equation of the form $q = r$, where q and r are polynomials with integer coefficients, possibly in several variables. A *system of Diophantine equations* is a finite set of Diophantine equations. For brevity, we refer to systems of Diophantine equations as D-systems. The notions of a *positive integer assignment* and a *positive integer solution* for a D-system are self explanatory.

We let DIO be the abstract logic whose sentences are the D-systems and whose models the positive integer assignments, with an assignment being a model of a D-system if, and only if, it is a solution.

EXAMPLE 5.4. We let 3SAT be the abstract logic whose formulas are the propositional formulas in 3-conjunctive normal form (3CNF), and whose models are the assignments for these formulas.

Translations between abstract logics can be seen as special types of reductions between the satisfiability problem of the two logics. With each sentence of the first logic we associate a sentence of the second in such a way that we can (easily) transfer models of the first sentence to models of the second and vice versa. The precise definition is rather technical:

DEFINITION 5.5. Let $\mathrm{A}_1 = (S_1, M_1)$ and $\mathrm{A}_2 = (S_2, M_2)$ be abstract logics. A *translation* between A_1 and A_2 is a triple $t = (t_1, t_2, t_3)$ of mappings such that:

(i) $t_1 : S_1 \to S_2$ is injective.
(ii) t_2 assigns a model $t_2(s, \boldsymbol{m}) \in M_2(t_1(s))$ to each pair $(s, \boldsymbol{m})$ where $s \in S_1,\ \boldsymbol{m} \in M_1(s)$.
(iii) t_3 assigns a model $t_3(s, \boldsymbol{n}) \in M_1(s)$ to each pair $(s, \boldsymbol{n})$ where $s \in S_1$, $\boldsymbol{n} \in M_2(t_1(s))$ in such a way that for all $s \in S_1,\ \boldsymbol{m} \in M_1(s)$ we have $t_3(s, t_2(s, \boldsymbol{m})) = \boldsymbol{m}$.

We say that t is a $(\mathsf{K}_1, \mathsf{K}_2, \mathsf{K}_3)$-translation, for complexity classes K_1, K_2, K_3, if t_1 is K_1-computable, t_2 is K_2-computable, and t_3 is K_3-computable.

We can compose translations in a natural way.

Here is a simple example of a translation:

EXAMPLE 5.6. There is a $(\mathsf{PTIME}, \mathsf{PTIME}, \mathsf{PTIME})$-translation $t = (t_1, t_2, t_3)$ from 3SAT to DIO. This is just a new formulation of a standard observation.

The complexity class PTIME/poly, *non-uniform* PTIME, has been introduced by Karp and Lipton [50]. A PTIME/poly-computation is a PTIME-computation that gets some additional information only depending on the size of the input structure for free. It is not important here to know the precise definition of PTIME/poly. Essentially, it suffices to know that it is considered very unlikely that NPTIME is contained in PTIME/poly, in particular because Karp and Lipton [50] proved that if NPTIME $\subseteq$ PTIME/poly then the polynomial hierarchy collapses.

THEOREM 5.7 (Grohe [30, 31]). *Let $k \geq 3$ and τ a vocabulary that contains at least one $(k-1)$-ary relation symbol. Then there is a* (PTIME, PTIME/poly, PTIME)*-translation from* DIO *to* INV$_k[\tau]$.

The complexity bounds in this theorem depend on the fact that we use a unary encoding for the integers occurring in a system of Diophantine equations. This dependence on the encoding disappears in the following applications of the theorem.

Matiyasevič's well-known theorem [58] states that there is no recursive function that associates a positive integer solution with each D-system. Thus Theorem 5.7 yields a negative answer to Questions 1 and 2:

COROLLARY 5.8. *Let $k \geq 3$ and τ a vocabulary that contains at least one $(k-1)$-ary relation symbol. Then I_τ^k has no recursive inverse $f : \mathcal{O}[\tau_{I^k}] \to \mathcal{F}[\tau]$.*

PROOF. Let $t = (t_1, t_2, t_3)$ be a (PTIME, PTIME/poly, PTIME)-translation from DIO to INV$_k[\tau]$. If $f : \mathcal{O}[\tau_{I^k}] \to \mathcal{F}[\tau]$ was a recursive inverse of I^k, then the mapping $D \mapsto t_3(D, f(t_1(D)))$ would be a recursive function that associates a positive integer solution with each D-system. ⊣

I cannot give a full proof of Theorem 5.7 here. Let me just give a brief outline. The first step is to settle the case $k = 3$:

LEMMA 5.9. *There is a* (PTIME, PTIME/poly, PTIME)*-translation from* DIO *to* INV$_3[\{E\}]$.

The central part of the proof of this lemma is the (probabilistic) construction of combinatorial gadgets called *scales*. They have two distinguished sets M and N, and their L^3-theory guarantees that M and N have the same size, without admitting any non-trivial definable relation between the two sets. We can use the scales to translate Diophantine equations to L^3-theories. For example, the equation $X = YZ$ is translated to an L^3-theory saying

- There are distinguished sets M_X, M_Y, M_Z, M_{YZ}.
- M_{YZ} is connected to M_Y and M_Z is such a way that its size is always the product of the sizes of M_Y and M_Z (via some binary relations).
- M_X and M_{YZ} form the sets M and N of a scale and thus always have the same size.

Thus the sizes of the sets M_X, M_Y, and M_Z in a model of the theory play the role of the values of X, Y, Z, respectively, in a positive integer solution of our equation. Scales are needed in this construction to avoid unwanted definable relations between different parts of a model of an L^3-theory encoding a complicated D-system. The crucial fact is that we can translate a D-system in such a way that the number of L^3-types of the resulting theory is bounded in terms of the size of the system, whereas the size of the models of the theory depends on the size of the positive integer solutions of the system.

The following lemma reduces the k-variable to the 3-variable case.

LEMMA 5.10. *Let $k \geq 4$ and τ a vocabulary that contains at least one $(k-1)$-ary relation symbol. Then there is a* (PTIME, PTIME/poly, PTIME)-*translation from* $\mathrm{INV}_3[\{E\}]$ *to* $\mathrm{INV}_k[\tau]$.

The proof of this lemma uses another probabilistic construction.

REMARK 5.11. The formal presentation of the results of this section differs considerably from the presentation in [30, 31]. It first appears in the full version of [30], which has not been published yet. It is available on my homepage (`http://sun2.mathematik.uni-freiburg.de/home/grohe`).

5.2. The canonization problem. A *canonization function* for an equivalence relation $\equiv$ on a class A is a $\gamma\colon A \to A$ such that

$$\text{(C1)} \qquad \forall a \in A\colon\ \gamma(a) \equiv a,$$

$$\text{(C2)} \qquad \forall a, b \in A\colon\ (a \equiv b \Longrightarrow \gamma(a) = \gamma(b)).$$

Note that if we have an encoding of the class A, then each canonization function for an equivalence relation on A gives rise to a complete invariant in the sense of Definition 4.8.

The definition of a canonization function needs further adjustment if we want to apply it to equivalence relations on classes of finite structures, since it is not entirely clear what equality of finite structures means—isomorphism is too coarse for our purposes. Suppose $\equiv$ is an equivalence relation on a class $\mathcal{C}$ of finite structures. Let α be an algorithm that computes a canonization function for $\equiv$. The input of α is a codeword $\boldsymbol{w} \in \mathcal{L}(\boldsymbol{A})$ of a τ-structure $\boldsymbol{A}$. The algorithm computes a word $\alpha(\boldsymbol{w})$, which is interpreted as the code of a τ-structure $\boldsymbol{A}' = \boldsymbol{e}^{-1}(\alpha(\boldsymbol{w}))\restriction_\tau$. (Note that up to isomorphism $\boldsymbol{A}'$ is the unique τ-structure with $\alpha(\boldsymbol{w}) \in \mathcal{L}(\boldsymbol{A}')$.) The condition we want to impose on the algorithm is the following

$$\text{(C2}'\text{)} \qquad \forall \boldsymbol{A}, \boldsymbol{B} \in \mathcal{C}\ \forall \boldsymbol{w} \in \mathcal{L}(\boldsymbol{A}), \boldsymbol{v} \in \mathcal{L}(\boldsymbol{B}) (\boldsymbol{A} \equiv \boldsymbol{B} \Longrightarrow \alpha(\boldsymbol{w}) = \alpha(\boldsymbol{v})).$$

Note that $\alpha(\boldsymbol{w}) = \alpha(\boldsymbol{v})$ is equivalent to the ordered structures $\boldsymbol{e}^{-1}(\alpha(\boldsymbol{w}))$ and $\boldsymbol{e}^{-1}(\alpha(\boldsymbol{v}))$ being isomorphic.

Letting

$$\mathcal{C}^{\leq} = \{\, (\boldsymbol{A}, \leq) \mid \boldsymbol{A} \in \mathcal{C},\ \leq \text{ is a linear order of } A \,\},$$

we end up with the following definition:

DEFINITION 5.12. Let $\equiv$ be an equivalence relation on a class $\mathcal{C}$ of τ-structures. A *canonization function* for $\equiv$ is a $\gamma : \mathcal{C} \to \mathcal{C}^{\leq}$ such that

$$\text{(SC1)} \qquad \forall \boldsymbol{A} \in \mathcal{C}\colon\ \boldsymbol{A} \equiv \gamma(\boldsymbol{A})\restriction_{\tau}.$$

$$\text{(SC2)} \qquad \forall \boldsymbol{A}, \boldsymbol{B} \in \mathcal{C}\colon\ (\boldsymbol{A} \equiv \boldsymbol{B} \Longrightarrow \gamma(\boldsymbol{A}) \cong \gamma(\boldsymbol{B})).$$

We say that $\equiv$ *admits* K*-canonization*, for a complexity class K, if there is a K-computable canonization function for $\equiv$.

As a corollary of Theorem 5.1 we obtain

COROLLARY 5.13 (Otto [62]). *For each vocabulary τ the equivalence relation $\equiv_{\mathsf{L}^2}$ on $\mathcal{F}[\tau]$ admits* PTIME*-canonization.*

PROOF. Let f be the inverse of I^2_τ obtained by Theorem 5.1. Then $f \circ I^2$ is a canonization function. ⊣

But again, this is the only positive result. For $k \geq 3$ we have:

THEOREM 5.14 (Grohe [31]). *Let $k \geq 3$ and τ a vocabulary that contains at least one $(k-1)$-ary relation symbol. Assume that* $\mathsf{NPTIME} \not\subseteq \mathsf{PTIME/poly}$*. Then the equivalence relation $\equiv_{\mathsf{L}^k}$ on $\mathcal{F}[\tau]$ does not admit* $\mathsf{PTIME/poly}$*-canonization.*

To prove this theorem, we combine our previous results with the following purely complexity theoretic Lemma 5.15.

Let $\equiv_{\mathrm{SAT}}$ be the following equivalence relation on the class of all pairs (Φ, α) where Φ is a 3CNF-formula and α a satisfying assignment for Φ:

$$(\Phi, \alpha) \equiv_{\mathrm{SAT}} (\Psi, \beta) \iff \Phi = \Psi.$$

LEMMA 5.15. *If there is a* $\mathsf{PTIME/poly}$*-computable canonization function for $\equiv_{\mathrm{SAT}}$ then*

$$\mathsf{NPTIME} \subseteq \mathsf{PTIME/poly}.$$

See [31] for a proof of this lemma.

PROOF (OF THEOREM 5.14). Suppose that γ is a PTIME-computable canonization function for $\equiv_{\mathsf{L}^k}$. Let $t = (t_1, t_2, t_3)$ be a ($\mathsf{PTIME}, \mathsf{PTIME/poly}, \mathsf{PTIME}$)-translation from 3SAT to INV$_k[\tau]$. Such a translation exists by Example 5.6 and Theorem 5.7. Then the function

$$(\Phi, \alpha) \mapsto (\Phi, t_3(\Phi, \gamma(t_2(\Phi, \alpha))))$$

is $\mathsf{PTIME/poly}$-computable canonization function for $\equiv_{\mathrm{SAT}}$. ⊣

It is worthwhile to compare our results on L^k-equivalence with the corresponding problems for first-order equivalence, which is the same as isomorphism. We have seen that deciding L^k-equivalence is PTIME-complete, that there is a PTIME-computable complete invariant for L^k-equivalence, and that unless the polynomial hierarchy collapses there is no PTIME-computable canonization function.

For isomorphism the situation looks quite different. Most people seem to believe that isomorphism cannot be decided in polynomial time. On the other hand, it is known that if deciding isomorphism is NPTIME-complete then the polynomial hierarchy collapses. Furthermore, the problem of finding a canonization function for isomorphism can be reduced to the problem of finding a complete invariant [36].

§**6. Finite variable logics with counting.** Although we can express important queries, such as connectivity of graphs, with few variables, our logics L^k are very weak in another respect: They cannot count very far. Remember Example 3.8, where we have seen that the logic $\mathsf{L}^k_{\infty\omega}$ cannot distinguish between two sets of cardinalities k and $k+1$, respectively, and Example 3.9, where we have seen that the class EVEN of all sets of even cardinality cannot be defined in the logic $\mathsf{L}^\omega_{\infty\omega}$.

On the other hand, such simple counting statements are of a very low computational complexity. So at least from a complexity theoretic point of view the expressive power of our finite variable logics is somehow unbalanced: On the one hand they can express all fixed-point definable queries, they can even express non-recursive queries, whereas on the other hand they fail to say that a set has even cardinality.

To overcome this mismatch, Immerman and Lander [45, 47, 49] augmented the logics by counting quantifiers such as $\exists^{\geq m}$. Recall the definition of the logic C^k (see page 350). We also consider the *infinitary k-variable logic with counting quantifiers* $\mathsf{C}^k_{\infty\omega}$ whose formulas are obtained by adding the rule

$$\frac{\Phi}{\bigvee \Phi} \quad \text{where } \Phi \text{ is a set of formulas}$$

to the rules forming C^k. Furthermore, we let $\mathsf{C}^\omega_{\infty\omega} = \bigcup_{k\geq 1} \mathsf{C}^k_{\infty\omega}$.

Obviously we have $\mathsf{L}^k \subseteq \mathsf{C}^k$ and $\mathsf{L}^k_{\infty\omega} \subseteq \mathsf{C}^k_{\infty\omega}$ for each $k \geq 1$, and thus $\mathsf{L}^\omega_{\infty\omega} \subseteq \mathsf{C}^\omega_{\infty\omega}$. The following simple example shows that these inclusions are strict and that the counting logics indeed can count.

EXAMPLE 6.1. For each $m \geq 1$ the C^1-sentence

$$\varphi_m = \exists^{\geq m} x\ (x = x) \wedge \neg\exists^{\geq m+1} x\ (x = x)$$

defines the class of all sets ($\emptyset$-structures) of size m, and for each $M \subseteq \mathbb{N}$ the $\mathsf{C}^1_{\infty\omega}$-sentence

$$\bigvee_{m\in M} \varphi_m$$

defines the class $\{\, (A) \mid |A| \in M \,\}$. Thus, in particular, EVEN is definable in $\mathsf{C}^1_{\infty\omega}$.

In the presence of an ordering we do not gain expressive power by adding counting quantifiers; it is not hard to see that $\mathsf{L}^{\omega}_{\infty\omega} = \mathsf{C}^{\omega}_{\infty\omega}$ on ordered structures. As a matter of fact, the main motivation for introducing the counting logics was to find a "substitute" for the order when trying to do descriptive complexity theory on arbitrary structures (see Section 8).

Although the finite-variable logics with counting are considerably more expressive than their counterparts without counting, in many other respects they behave very similarly. We have already seen an example of such a similar behavior in Subsection 2.3: The complexity of MODEL CHECKING is the same for the finite variable logics with and without counting.

6.1. Inflationary fixed-point logic with counting. We want to extend inflationary fixed-point logic by the ability to speak about cardinalities of definable sets and do some computations with them.

With each finite structure $\boldsymbol{A}$ we associate a two-sorted structure $\boldsymbol{A}^{\#}$. Its restriction to the first sort is $\boldsymbol{A}$. The second sort consists of the set $[0, |A|]$ of natural numbers which is equipped with the natural ordering $\leqslant$. Thus we may write $\boldsymbol{A}^{\#} = \boldsymbol{A} \mathbin{\dot\cup} ([0, |A|], \leqslant)$. We usually refer to the elements of the first sort of $\boldsymbol{A}^{\#}$ as "points" and to the elements of the second sort as "numbers".

We can now study inflationary fixed-point logic on these two-sorted structures. As a convention, we use symbols x, y, z (and variants) for variables ranging over points and μ, ν (and variants) for variables ranging over numbers. If we do not want to fix the type of variable we refer to we use u, v, w (and variants). Relation variables may range over both sorts. We denote the set of all two-sorted IFP-formulas by $\mathsf{IFP}^{\#}$.

We are mainly interested in those $\mathsf{IFP}^{\#}$-formulas whose free variables only range over points. Each such formula defines a query on the class of one-sorted structures of the respective vocabulary. When comparing the expressive power of $\mathsf{IFP}^{\#}$ with that of other logics, we always refer to the class of queries definable by $\mathsf{IFP}^{\#}$-formulas whose free variables are restricted to points.

Nevertheless, we have $\mathsf{IFP} \subset \mathsf{IFP}^{\#}$. For example, the class of all $\emptyset$-structures whose universe has even cardinality can be defined by an $\mathsf{IFP}^{\#}$-sentence, but not by an IFP-sentence.

Inflationary fixed-point logic with counting $\mathsf{IFP{+}C}$ is an extension of $\mathsf{IFP}^{\#}$ that also lives in the two-sorted framework described above. The class of $\mathsf{IFP{+}C}$-formulas is obtained by adding a new formula-formation rule to the rules for $\mathsf{IFP}^{\#}$:

> If $\varphi(x)$ is a formula, x is a point-variable, and μ a number-variable, then we may form the new formula $\exists^{=\mu} x\varphi(x)$.

The free variables of $\exists^{=\mu} x\varphi(x)$ are μ and the free variables of φ other than x. The semantics is defined by

$$\boldsymbol{A}^{\#} \models \exists^{=i} x\varphi(x) \iff i \text{ is the number of } a \in A \text{ such that } \boldsymbol{A}^{\#} \models \varphi(a).$$

Note that $i \in [0, |A|]$, the number-part of $\boldsymbol{A}^{\#}$.

For example, the following IFP+C-formula says that a vertex x of a graph has even valence:

$$\exists \mu([\mathrm{IFP}_{\mu,X}\, \mu = 0 \vee \exists \nu \in X\ \mu = \nu + 2]\mu \wedge \exists^{=\mu} yExy),$$

where $\mu = 0$ and $\mu = \nu + 2$ are abbreviations with the obvious meaning. Using the pebble game (on the two-sorted structure $\boldsymbol{A}^{\#}$) it can easily be seen that this query is not definable in $\mathsf{IFP}^{\#}$. Thus we have $\mathsf{IFP}^{\#} \subset \mathsf{IFP{+}C}$.

Fixed-point logic with counting has been introduced by Immerman [47]. In the form presented here it is due to Grädel and Otto [25]. Though it may seem artificial at first sight, the logic has turned out to be quite robust. Grädel and Otto [25] studied several different formalizations of the concept "inductive definability with counting" and proved them all to be equivalent.

Proposition 6.2 (Otto [63]).

$$\mathsf{IFP{+}C} \subseteq \mathsf{C}^{\omega}_{\infty\omega}.$$

The proof of this result is simple, but a bit tedious (see [19]). One first replaces a fixed-point formula by an infinitary formula which still may contain subformulas of the form $\exists^{=\mu} x\varphi(x)$ and thus lives in the two-sorted framework (as in Proposition 3.4) and then eliminates quantification over number-variables by infinite disjunctions and conjunctions.

The analogue of Proposition 3.2 is also true:

Proposition 6.3. *For each* IFP+C-*formula* φ *there is a* PTIME-*algorithm that checks whether a given structure is a model of* φ.

6.2. The expressive power. There is a straightforward extension of the k-pebble game to a game characterizing equivalence in the logic $\mathsf{C}^{k}_{\infty\omega}$ [45, 49].

The connection between the k-*bijective game*, to be introduced next, and the counting logics is less obvious. Hella [39] originally introduced a slightly different version of the game to characterize equivalence in the extension $\mathsf{L}^{k}_{\infty\omega}(\mathbf{Q}_1)$ of the logic $\mathsf{L}^{k}_{\infty\omega}$ by all unary Lindström-quantifiers.

Definition 6.4. Let $k \geq 1$ and τ a vocabulary. The k-*bijective game* is played by two players, a *spoiler* and a *duplicator*, on a pair $\boldsymbol{AB}$ of τ-structures.

A *position* of the game is a subset of $A \times B$ of size at most k. If $|A| = |B|$, the game starts in an initial position and consists of a sequence of rounds. If $|A| \neq |B|$, the spoiler wins immediately.

In each round of the game the spoiler first removes a pair from the current position if its size is k. Let P be the resulting position (of size at most $(k-1)$). Now the duplicator defines a bijection β between A and B. The spoiler chooses an $a \in A$ and the new position is $P \cup \{a\beta(a)\}$.

The duplicator *wins* the game if $|A| = |B|$ and each position P that occurs is a partial isomorphism between $\boldsymbol{A}$ and $\boldsymbol{B}$.

THEOREM 6.5 (Hella [40]). *Let $k \geq 1$ and τ a vocabulary. Furthermore, let $\boldsymbol{A}, \boldsymbol{B}$ be τ-structures and $\bar{a} = a_1 \dots a_l \in A^l$, $\bar{b} = b_1 \dots b_l \in B^l$ l-tuples, for an $l \leq k$. Then we have*:

(i) $\boldsymbol{A} \equiv_{\mathsf{C}^k_{\infty\omega}} \boldsymbol{B}$ *if, and only if, the duplicator has a winning strategy for the k-bijective game on $\boldsymbol{AB}$ with initial position $\emptyset$.*

(ii) $\mathrm{tp}_{\mathsf{C}^k_{\infty\omega}}(\bar{a}) = \mathrm{tp}_{\mathsf{C}^k_{\infty\omega}}(\bar{b})$ *if, and only if, the duplicator has a winning strategy for the k-bijective game on $\boldsymbol{AB}$ with initial position $\{a_1b_1, \dots, a_lb_l\}$.*

The proof of this theorem is not as straightforward as the proof of the analogous result for the k-pebble game (Theorem 3.7). The next example illustrates the basic idea.

EXAMPLE 6.6. Let $\boldsymbol{A}$ be a τ-structure and $\bar{a} \in A^k$. We define a C^k-formula $\chi^{k,1}_{\boldsymbol{A},\bar{a}}(\bar{y})$ such that for all τ-structures $\boldsymbol{B}$ and $\bar{b} \in B^k$ we have

$$\boldsymbol{B} \models \chi^{k,1}_{\boldsymbol{A},\bar{a}}(\bar{b}) \iff \text{The spoiler cannot win the } k\text{-bijective game on } \boldsymbol{AB} \text{ with initial position } \{a_1b_1, \dots, a_kb_k\} \text{ in the first round.}$$

Our formula has to say that for all $i \in [1,k]$ there is a bijection $\beta \colon A \to B$ such that for all $a \in A$ the tuples $\bar{a}\frac{a}{i}$ and $\bar{b}\frac{\beta(a)}{i}$ have the same atomic type. For each $i \in [1,k]$ and atomic k-type θ of vocabulary τ, let $n(\theta, i)$ be the number of $a \in A$ such that $\bar{a}\frac{a}{i}$ realizes θ. What we actually say is that for each $i \in [1,k]$ and each atomic type θ there are precisely $n(i,\theta)$ tuples of the form $\bar{b}\frac{b}{i}$ that realize θ. This clearly implies the existence of the desired bijection. Hence we let

$$\chi^{k,1}_{\boldsymbol{A},\bar{a}}(\bar{y}) = \bigwedge_{i=1}^{k} \bigwedge_{\theta \text{ atomic } k\text{-type}} \exists^{=n(i,\theta)} y\ \theta(\bar{y}\tfrac{y}{i}).$$

Let us now turn to some inexpressibility results. Example 3.10 can easily be extended to the counting case:

EXAMPLE 6.7. The class of connected graphs is not definable by a $\mathsf{C}^2_{\infty\omega}$-sentence. I leave it to the reader to prove this by showing that the duplicator has a winning strategy for the 2-bijective game on a graph $\boldsymbol{A}$ consisting of a cycle of length 6 and a graph $\boldsymbol{B}$ consisting of two cycles of length 3.

Note that this shows that there is an L^3-sentence not equivalent to any C^2-sentence.

EXAMPLE 6.8. Recall the definition of a projective plane from Subsection 2.1. We are now ready to prove that two projective planes of order n are C^3-equivalent. We use a few basic facts on projective planes that can be found in any book on combinatorics (for example [10]).

So let $\boldsymbol{P} = (P, L, I)$ and $\boldsymbol{Q} = (Q, M, J)$ be two projective planes of order n. I claim that for all $x_1, x_2 \in P \cup L$, $y_1, y_2 \in Q \cup M$ such that $\mathrm{atp}(x_1x_2) = \mathrm{atp}(y_1y_2)$ the duplicator has a winning strategy for the 3-bijective game on

$\boldsymbol{PQ}$ with initial position $\{x_1y_1, x_2y_2\}$. Clearly this suffices to prove that $\boldsymbol{P} \equiv_{\mathsf{C}^3} \boldsymbol{Q}$.

So let x_1, x_2, y_1, y_2 be such elements. We show how to define a bijection $\beta\colon P \cup L \to Q \cup M$ such that for all $x_3 \in P \cup L$ we have

$$\mathrm{atp}(x_1x_2x_3) = \mathrm{atp}(y_1y_2\beta(x_3)).$$

Suppose first that $x_1, x_2 \in P$. Without loss of generality we can assume that $x_1 \neq x_2$. Then there is a unique line l_1 incident to both x_1 and x_2, there are precisely n lines $l_2, \dots, l_{n+1}$ that are incident to x_1 but not x_2 and n lines l_{n+2}, $\dots$, l_{2n+1} incident to x_2 but not x_1. The remaining lines $l_{2n+2}, \dots, l_{n^2+n+1}$ are neither incident to x_1 nor x_2. In (Q, M, J) the situation is analogous: There is one line M_1 incident to both y_1 and y_2, n lines m_2, $\dots$, m_{n+1} incident to y_1 but not y_2, n lines $m_{n+2}, \dots, m_{2n+1}$ incident to y_2 but not y_1, and $n^2 - n$ lines m_{2n+2}, $\dots$, m_{n^2+n+1} incident to neither y_1 nor y_2. We let $\beta(l_i) = m_i$ for $i \in [1, n^2 + n + 1]$, this clearly preserves the incidences with x_1, x_2, y_1, y_2. To define β on the points, we let $\beta(x_1) = y_1$, $\beta(x_2) = y_2$, and extend it arbitrarily to a bijection.

We can argue similarly if $x_1 \in P$, $x_2 \in L$, and the rest follows by duality.

We can now deduce that there is a first-order sentence φ that is not equivalent to any $\mathsf{C}^3_{\infty\omega}$-sentence. φ simply says that a projective plane is Desarguian. This sentence cannot be equivalent to a $\mathsf{C}^3_{\infty\omega}$-sentence, because for each order p^r, where p is prime and $r \geq 2$, both Desarguian and non-Desarguian projective planes are known. But we have just seen that they are indistinguishable in the logic $\mathsf{C}^3_{\infty\omega}$.

I do not know any natural examples of inexpressibility results for $\mathsf{C}^4_{\infty\omega}$ (except, of course, queries on structures with 5-ary relations). However, Cai, Fürer, and Immerman proved the following:

THEOREM 6.9 (Cai, Fürer, Immerman [9]). *For each $k \in \mathbb{N}$ there exist non-isomorphic graphs $\boldsymbol{G}_k, \boldsymbol{H}_k$ of size $O(k)$, valence* 3, *and color class size* 4 *such that $\boldsymbol{G}_k$ and $\boldsymbol{H}_k$ satisfy the same $\mathsf{C}^k_{\infty\omega}$-sentences.*

Using similar methods as Cai et al., Otto proved the following:

THEOREM 6.10 (Otto [64]). *For each $k \geq 1$ there is an L^{k+1}-sentence not equivalent to any $\mathsf{C}^k_{\infty\omega}$-sentence.*

6.3. The basic theory. The theory of Section 4 can be transferred to the counting case without any problems. We only state the most important results. Example 6.6 gives a hint how to transfer the proofs. The results of this subsection are due to Otto [63].

LEMMA 6.11. *Let τ be a vocabulary and $k \geq 1$.*

(i) *The $2k$-ary query $\approx_k$ that associates the relation*

$$\approx_k^{\boldsymbol{A}} = \{\, \bar{a}\bar{b} \in A^{2k} \mid \mathrm{tp}_{\mathsf{C}^k_{\infty\omega}}(\bar{a}) = \mathrm{tp}_{\mathsf{C}^k_{\infty\omega}}(\bar{b}) \,\}$$

with each τ-structure $\boldsymbol{A}$ is definable in IFP+C.

(ii) *There is an* IFP+C*-definable* $2k$*-ary query* $\precsim_k$ *that associates with each* τ*-structure* $\boldsymbol{A}$ *a pre-order* $\precsim_k^{\boldsymbol{A}}$ *on* A^k *whose classes are the equivalence classes of* $\approx_k^{\boldsymbol{A}}$.

Furthermore, $\precsim_k$ *orders the* $\mathsf{C}^k_{\infty\omega}$*-types in the same way in all structures* (*see Lemma* 4.2 *for a precise formulation*).

Note that by Proposition 6.3 the queries $\approx_k$ and $\precsim_k$ are PTIME-computable.

LEMMA 6.12. *Let* $k \geq 1$, τ *a vocabulary,* $\boldsymbol{A}$ *a* τ*-structure, and* $\bar{a} \in A^k$. *Then there is a* C^k*-formula* $\chi^k_{\boldsymbol{A},\bar{a}}(\bar{y})$ *such that for all* τ*-structures* $\boldsymbol{B}$ *and* $\bar{b} \in B^k$ *we have*

$$\boldsymbol{B} \models \chi^k_{\boldsymbol{A},\bar{a}}(\bar{b}) \iff \mathrm{tp}_{\mathsf{C}^k}(\bar{a}, \boldsymbol{A}) = \mathrm{tp}_{\mathsf{C}^k}(\bar{b}, \boldsymbol{B}).$$

COROLLARY 6.13. *The* C^k*-theory of a structure* $\boldsymbol{A}$ *is finitely axiomatizable in* C^k.

COROLLARY 6.14. *Let* $k \geq 1$. *Each* $\mathsf{C}^k_{\infty\omega}$*-sentence is equivalent to a* C^k*-theory.*

COROLLARY 6.15. *Let* $\boldsymbol{A}$, $\boldsymbol{B}$ *be* τ*-structures and* $\bar{a} \in A^l$, $\bar{b} \in B^l$, *where* $l \leq k$. *Then the following statements are equivalent*:

(i) $\mathrm{tp}_{\mathsf{C}^k}(\bar{a}, \boldsymbol{A}) = \mathrm{tp}_{\mathsf{C}^k}(\bar{b}, \boldsymbol{B})$.
(ii) $\mathrm{tp}_{\mathsf{C}^k_{\infty\omega}}(\bar{a}, \boldsymbol{A}) = \mathrm{tp}_{\mathsf{C}^k_{\infty\omega}}(\bar{b}, \boldsymbol{B})$.

6.4. The invariant. We are now going to define an invariant J^k for $\equiv_{\mathsf{C}^k}$. To store the counting information the C^k-theory of a structure $\boldsymbol{A}$ contains, $J^k(\boldsymbol{A})$ will be a two-sorted structure that has a point and a number sort (just like $\boldsymbol{A}^{\#}$).

For a vocabulary τ we let τ_{J^k} be the vocabulary consisting of the binary relation symbol $\leq$ of sort 'point $\times$ point', a binary relation symbol $\leqslant$ of type 'number $\times$ number', ternary relation symbols $F_1, \ldots, F_k$ of type

'point $\times$ point $\times$ number',

and a unary relation symbol P_θ of sort 'point' for each atomic k-type θ of vocabulary τ.

With each τ-structure $\boldsymbol{A}$ we associate a τ_{J^k}-structure $\boldsymbol{J} = J^k(\boldsymbol{A})$ defined as follows:

- The universe of the point sort is $A^k/_{\approx_k}$, and the universe of the number sort is $[0, |A|]$.
- $\leq^{\boldsymbol{J}}$ is the order on the points induced by $\precsim_k$.
- $\leqslant^{\boldsymbol{J}}$ is the natural order on the numbers.
- For $i \leq k$, the relation $F_i^{\boldsymbol{J}}$ is defined by:

$$F_i^{\boldsymbol{J}}(\bar{a}/_{\approx_k})(\bar{a}'/_{\approx_k})j \iff j = \left|\{\, a' \in A \mid \bar{a}' \approx_k \bar{a}\tfrac{a'}{i} \,\}\right|$$

(for $\bar{a}, \bar{a}' \in A^k$ and $j \in [0, |A|]$).

- For each atomic k-type θ, the relation $P_\theta^{\boldsymbol{J}}$ is defined by

$$P_\theta^{\boldsymbol{J}}(\bar{a}/_{\approx_k}) \iff \operatorname{atp}(\bar{a}) = \theta$$

(for $\bar{a} \in A^k$).

Analogously to the case without counting, it can be proved that $J^k : \mathcal{F} \to \mathcal{O}$ is well defined. J_τ^k denotes the restriction of J^k to $\mathcal{F}[\tau]$. The analogue of Theorem 4.10 holds:

THEOREM 6.16 (Otto [63]). *Let τ be a vocabulary and $k \geq 1$. Then the following holds*:

(i) *For all τ-structures $\boldsymbol{A}$ and $\boldsymbol{B}$ we have*

$$\boldsymbol{A} \equiv_{\mathsf{C}^k} \boldsymbol{B} \iff J^k(\boldsymbol{A}) \cong J^k(\boldsymbol{B}).$$

(ii) *J_τ^k is* PTIME-*computable.*

COROLLARY 6.17. *Let $k \geq 1$ and τ a vocabulary. Then there is a* PTIME-*computable complete invariant for the equivalence relation $\equiv_{\mathsf{C}^k}$ on $\mathcal{F}[\tau]$.*

We use similar conventions for J^k as those introduced for I^k on page 367 after the statement of Corollary 4.11. In particular, we refer to J^k as the C^k-*invariant*.

It is an immediate consequence of Corollary 6.17 that C^k-equivalence can be decided in PTIME. This result is due to Immerman and Lander [49]. As in the case without counting we have:

THEOREM 6.18 (Grohe [29]). $\mathsf{C}^k[\tau]$-EQUIVALENCE *is* PTIME-*complete under quantifier-free reductions.*

6.5. Inverting the invariants. For the 2-variable case, Otto [62] proved the analogue of Theorem 5.1.

THEOREM 6.19 (Otto [62]). *For each vocabulary τ there is a* PTIME-*computable inverse for J_τ^2.*

COROLLARY 6.20. *For each vocabulary τ the restriction of $\equiv_{\mathsf{C}^2}$ to $\mathcal{F}[\tau]$ admits* PTIME-*canonization*.

Let $k \geq 3$ in the following. We are still in a much better position when it comes to inverting J^k than we were with I^k. It is easy to see that $J^k(\mathcal{F}[\tau])$ is recursive. As a matter of fact, the set is in NPTIME. The reason is that the invariant $J^k(\boldsymbol{A})$ of a τ-structure $\boldsymbol{A}$ contains information on the cardinality of $\boldsymbol{A}$. To find out whether an arbitrary $\boldsymbol{J} \in \mathcal{O}[\tau_{J^k}]$ belongs to $J^k(\mathcal{F}[\tau])$ we can just guess a τ-structure $\boldsymbol{A}$ of the appropriate cardinality and then verify in PTIME that $\boldsymbol{J} = J^k(\boldsymbol{A})$.

OPEN QUESTION 6.21. *Let $k \geq 3$ and τ a vocabulary that contains at least one binary relation symbol. Is there a* PTIME-*computable inverse $f : \mathcal{O}[\tau_{J^k}] \to \mathcal{F}[\tau]$ of J_τ^k?*

Note that a positive answer to this question for $k = 3$ and $\tau = \{P, L, I\}$ would yield a PTIME-algorithm deciding whether there is a projective plane of a given order. However, I believe that the answer to the question is negative, that is, that J^k has no PTIME-computable inverse for $k \geq 3$.

Open Question 6.22. *Let $k \geq 3$ and τ a vocabulary that contains at least one binary relation symbol. Is there a* PTIME*-computable canonization function for* C^k*-equivalence on the class of all finite τ-structures*?

6.6. Isomorphism and $\mathbf{C}^k$-equivalence. As we have discussed earlier, classes on which the "combinatorial notion" C^k-equivalence coincides with the "structural notion" isomorphism are particularly interesting, one reason being of course that C^k-equivalence can be tested in polynomial time.

Recall that by Corollary 2.5 on almost all graphs C^2-equivalence coincides with isomorphism. But of course one may argue that almost all interesting graphs are not among these. So let us consider further examples that may be more relevant:

Proposition 6.23 (Immerman, Lander [49]). *On the class of trees,* C^2*-equivalence and isomorphism coincide.*

Proof. Recall that a *tree* is an undirected, loop-free graph that is connected and cycle-free. The *leaves* of a tree are the vertices of valence 1.

Let $\boldsymbol{T} = (T, E^{\boldsymbol{T}})$ be a tree. Then the 0th *level* is defined to be the set of all leaves, and for $i \geq 1$ the ith level is the set of all vertices $a \in T$ such that all neighbors of a except maybe one belong to a lower level (see Figure 1).

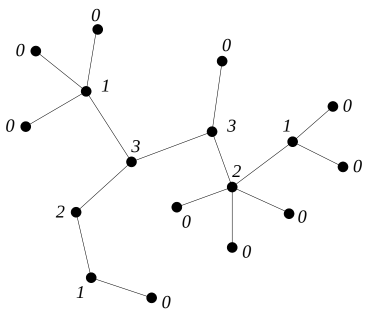

Figure 1. A tree with four levels

Clearly each level of a tree is definable in $\mathsf{C}^2_{\infty\omega}$. The *children* of a vertex a are the vertices of a lower level that are adjacent to a, and the *subtree* with root a is the substructure consisting of a, its children, their children, et cetera.

We prove that the spoiler has a winning strategy for the 2-bijective game on a pair $\boldsymbol{UT}$ of non-isomorphic trees. The duplicator starts the first round of the game by defining a bijection $\beta\colon U \to T$. There must be an $a \in T$ such that the subtree of $\boldsymbol{T}$ with root a is not isomorphic to the subtree with root $\beta(a)$. The spoiler picks a, and the position after the first round is $P_1 = \{a\beta(a)\}$.

In the second round, the duplicator defines a bijection γ. If it does not map a to $\beta(a)$ and the children of a to the children of $\beta(a)$, then the spoiler wins, since the set of children of a vertex is definable with two variables. If it does, there must be a child b of a such that the subtrees with root b and $\gamma(b)$ are not isomorphic. The spoiler picks b, and the new position is $P_2 = \{a\beta(a), b\gamma(b)\}$. In the third round the spoiler removes $a\beta(a)$ from position P_2. The duplicator chooses a bijection δ. Now the game continues as in the second round, and, unless the spoiler wins, after that round it will be in a position $\{b\gamma(b), c\delta(c)\}$ where the subtrees with roots c and $\delta(c)$ are not isomorphic. The game goes on like this, and since the level decreases each round the spoiler eventually wins. $\dashv$

We state three further results without proof.

THEOREM 6.24.

(i) (*Grohe* [32]) *There is a $k \geq 1$ such that on the class of planar graphs, isomorphism and C^k-equivalence coincide.*

(ii) (*Grohe, Mariño* [33]) *Let $k \geq 1$. On graphs of tree-width at most k, isomorphism and C^{k+3}-equivalence coincide.*

(iii) (*Immerman, Lander* [49]) *On graphs of color class size at most 3, isomorphism and L^3-equivalence coincide.*

The previous results can all be extended to colored graphs of the respective classes. Immerman and Lander [49] observed that this implies that isomorphism on these classes admits PTIME-canonization. One way to make this precise is the following, where we model colored graphs by graphs with a pre-order. (Thus we consider the set of colors as ordered.)

PROPOSITION 6.25. *Let $k \geq 1$ and $\mathcal{C}$ a class of structures such that isomorphism coincides with C^k-equivalence on*

$$\mathcal{C}^{\preceq} = \{ (\boldsymbol{A}, \preceq) \mid \boldsymbol{A} \in \mathcal{C}, \preceq \text{ is a pre-order on } A \}.$$

Then isomorphism on $\mathcal{C}$ admits PTIME-canonization.

This gives rise to PTIME-canonization algorithms for isomorphism on trees, planar graphs, and graphs of bounded tree-width. The next section shows that on these classes we actually have more.

§7. Definable canonization. For a formula $\varphi(\bar{v})$ with an l-tuple $\bar{v}$ of free variables and a structure $\boldsymbol{A}$ we let

$$\varphi(\bar{v})^{\boldsymbol{A}} = \{ \bar{a} \in A^l \mid \boldsymbol{A} \models \varphi(\bar{a}) \}.$$

Let L be a logic, $l \geq 1$, and τ, $\sigma = \{R_1, \dots, R_m\}$ vocabularies, where R_i is r_i-ary. An *l-dimensional L-interpretation* of σ in τ is a sequence

$$\Phi = (\varphi_{\text{uni}}(\bar{v}), \varphi_{R_1}(\bar{v}_1, \dots, \bar{v}_{r_1}), \dots, \varphi_{R_m}(\bar{v}_1, \dots, \bar{v}_{r_m}))$$

of L-formulas, where $\bar{v}$ and the $\bar{v}_i$ are l-tuples.

For each τ-structure $\boldsymbol{A}$ we let $\Phi^{\boldsymbol{A}}$ be the σ-structure

$$(\varphi_{\text{uni}}(\bar{v})^{\boldsymbol{A}}, \varphi_{R_1}(\bar{v}_1, \dots, \bar{v}_{r_1})^{\boldsymbol{A}} \cap (\varphi_{\text{uni}}(\bar{v})^{\boldsymbol{A}})^{r_1}, \dots, \\ \varphi_{R_m}(\bar{v}_1, \dots, \bar{v}_{r_m})^{\boldsymbol{A}} \cap (\varphi_{\text{uni}}(\bar{v})^{\boldsymbol{A}})^{r_m}).$$

Here we view $\varphi_{R_i}(\bar{v}_1, \dots, \bar{v}_{r_i})^{\boldsymbol{A}}$ as an r_i-ary relation on A^l, rather than an $r_i l$-ary relation on A.

If $\mathsf{L} = \mathsf{IFP}^{\#}$ or $\mathsf{IFP{+}C}$ we admit variables of both sorts to appear in the tuples $\bar{v}$ and $\bar{v}_i$. Thus in this case the universe of the structure $\Phi^{\boldsymbol{A}}$ is a subset of $(A \cup [0, |A|])^l$ of the appropriate type.

REMARK 7.1. The concept of L-interpretations can be refined in various ways, for example by defining equality as a congruence relation on $\Phi^{\boldsymbol{A}}$, by admitting parameters, or by admitting constants and function symbols in the vocabulary (see, for example, [43]). This also leads to refined notions of definable canonization which are useful in some situations.

Recall the definition of a canonization function for an equivalence relation on a class of structures (Definition 5.12).

DEFINITION 7.2. Let L be a logic and $\equiv$ an equivalence relation on a class $\mathcal{C}$ of τ-structures. We say that $\equiv$ *admits* L-*definable canonization* if there is an L-interpretation Φ of $\tau \cup \{\leq\}$ in τ such that the mapping $\boldsymbol{A} \mapsto \Phi^{\boldsymbol{A}}$ is a canonization function for $\equiv$.

If isomorphism on $\mathcal{C}$ admits L-definable canonization, we just say that $\mathcal{C}$ admits L-definable canonization.

Note that if $\equiv$ is isomorphism on $\mathcal{C}$, then it suffices to require that for all $\boldsymbol{A} \in \mathcal{C}$ we have $\boldsymbol{A} \cong \Phi^{\boldsymbol{A}}\restriction_{\tau}$, because condition (SC2) (of Definition 5.12) is automatically true in this case.

The easiest way to obtain definable canonization is by defining an order. We say that a formula $\varphi(x, y, \bar{z})$ (with *parameters* $\bar{z}$) *defines an order* on a class $\mathcal{C}$ of structures if for all $\boldsymbol{A} \in \mathcal{C}$ there is a tuple $\bar{c}$ such that the binary relation

$$\{\, ab \in A^2 \mid \boldsymbol{A} \models \varphi(a, b, \bar{c}) \,\}$$

is a linear order of A.

PROPOSITION 7.3. *Let* L *be* IFP *or* IFP+C.[5] *If there is an* L-*formula* (*possibly with parameters*) *defining an order on a class* $\mathcal{C}$ *then* $\mathcal{C}$ *admits* L-*definable canonization*.

PROOF. The statement is obvious if the formula φ defining an order on $\mathcal{C}$ has no parameters. If it has parameters, it actually associates a family of ordered copies of $\boldsymbol{A}$ with each structure $\boldsymbol{A} \in \mathcal{C}$. These ordered copies can be

[5]The Proposition holds true for all regular logics, but we are only interested in these two here.

ordered by the lexicographical order of their codewords, and actually this order among the structures is definable. So we can simply let the first ordered copy with respect to this ordering be the canonical one. ⊣

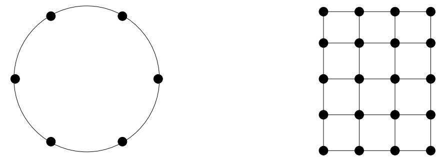

FIGURE 2. A cycle and a grid.

For example, on the class of undirected cycles and on the class of grids (see Figure 2) an order is IFP-definable with two parameters.

A more interesting example follows from the results of Subsection 2.2.

THEOREM 7.4 (Hella, Kolaitis, Luosto [41]). *There is a class $\mathcal{C}$ of graphs with $\mu(\mathcal{C}) = 1$ such that there is an* IFP+C-*definable order on $\mathcal{C}$. Thus in particular, $\mathcal{C}$ admits* IFP+C-*definable canonization.*

PROOF. By Lemma 6.11 (ii) there is an IFP+C-formula $\varphi(x, x', y, y')$ defining the pre-order $\precsim_2$ of the C^2-types. Recall that by Proposition 2.3 and Theorem 2.4 there is a class $\mathcal{C}$ of graphs with $\mu(\mathcal{C}) = 1$ such that for all $\boldsymbol{A} \in \mathcal{C}$, $a \neq b \in A$ we have $\mathrm{tp}_{\mathsf{C}^2}(a) \neq \mathrm{tp}_{\mathsf{C}^2}(a)$. On this class the formula $\psi(x, y) = \varphi(x, x, y, y)$ defines an order. ⊣

It is important to realize the difference between definable canonization and definability of an order. It is illustrated by the following example.

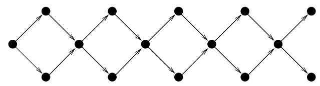

FIGURE 3. A chain with five crossnodes.

EXAMPLE 7.5. A chain is an $\{E\}$-structure as given in Figure 3. We call the nodes of out-degree 2 the *cross-nodes* and the nodes of in-degree 1 the *link-nodes* of a chain. Let $\mathcal{C}$ denote the class of all chains.

Obviously, no order is definable on the class $\mathcal{C}$, not even with (a finite number of) parameters, because there are automorphisms switching two corresponding link-nodes and leaving the rest fixed.

But $\mathcal{C}$ admits IFP-definable canonization. To prove this, let $\gamma(x)$ be a formula defining the cross-nodes. Let

$$\psi(x, y) = \gamma(x) \wedge \gamma(y) \wedge \exists z(Exz \wedge Ezy)$$

and let $\chi(x, y)$ define the reflexive transitive closure of $\psi(x, y)$. Note that $\chi(x, y)$ defines an order on the cross-nodes. Without loss of generality we only consider chains with at least three cross-nodes; let $\gamma_1(x)$, $\gamma_2(x)$, $\gamma_3(x)$ be formulas defining the first, second, third cross-node, respectively, with respect to the order defined by χ.

The formulas below form a 2-dimensional IFP-interpretation of $\{E, \leq\}$ in $\{E\}$ that defines a canonization function:

$$\begin{aligned}
\varphi_{\mathrm{uni}}(x, y) &= (\gamma_1(x) \vee \gamma_2(x) \vee \gamma_3(x)) \wedge \gamma(y),\\
\varphi_E(x, y, x', y') &= (\gamma_1(x) \wedge \gamma_2(x') \wedge y = y')\\
&\quad \vee (\gamma_1(x) \wedge \gamma_3(x') \wedge y = y')\\
&\quad \vee (\gamma_2(x) \wedge \gamma_1(x') \wedge \psi(y, y'))\\
&\quad \vee (\gamma_3(x) \wedge \gamma_1(x') \wedge \psi(y, y')),\\
\varphi_{\leq}(x, y, x', y') &= \bigvee_{i<j\leq 3} (\gamma_i(x) \wedge \gamma_j(x'))\\
&\quad \vee \bigvee_{i\leq 3} (\gamma_i(x) \wedge \gamma_i(x') \wedge \chi(y, y')).
\end{aligned}$$

PROPOSITION 7.6. *The class of trees admits* IFP+C-*definable canonization*.

PROOF. We define an IFP+C-formula $\varphi_E(\mu, \nu)$ such that for all trees $\boldsymbol{T}$ the structure

$$\Big([1, |T|], \varphi_E(\mu, \nu)^{\boldsymbol{T}}\Big)$$

is isomorphic to $\boldsymbol{T}$. This clearly yields definable canonization via

$$\Phi = (\varphi_{\mathrm{uni}}(\mu), \varphi_E(\mu, \nu), \varphi_{\leq}(\mu, \nu))$$

where

$$\varphi_{\mathrm{uni}}(\mu) = \exists \nu\ \nu < \mu \quad \text{and} \quad \varphi_{\leq}(\mu, \nu) = (\mu \leqslant \nu).$$

Let $\boldsymbol{T}$ be a tree. For $t \in T$, let $\boldsymbol{S}(t)$ be the subtree of $\boldsymbol{T}$ with root t. We inductively define a ternary relation X of type 'point×number×number' such that for all vertices $t \in T$ the $\{E\}$-structure

$$\boldsymbol{X}(t) = \big([1, |S(t)|], \{\, ij \in [1, |S(t)|]^2 \mid Xtij \,\}\big)$$

is isomorphic to $\boldsymbol{S}(t)$.

For all t, let X_t denote the set of all pairs ij such that $Xtij$. The induction is on the level of the vertex t. If t is a leaf then we just let X_t be empty.

If t is of level at least one, let $t_1, \dots, t_l$ be its children. Suppose that X_{t_i} is defined for all $i \in [1, l]$. Recall that $\leqslant$ denotes the natural ordering on the numbers and let $\boldsymbol{X}_i = (\boldsymbol{X}(t_i), \leqslant)$. The $\boldsymbol{X}_i$ are ordered structures, and we can compute their codewords $\boldsymbol{e}(\boldsymbol{X}_i)$. Let $\boldsymbol{w}_1 < \cdots < \boldsymbol{w}_m$ be a lexicographical enumeration of the set $\{\, \boldsymbol{e}(\boldsymbol{X}_i) \mid i \in [1, l] \,\}$. Note that $m \leq l$. Let n_i be the number of occurrences of $\boldsymbol{w}_i$ among the $\boldsymbol{e}(\boldsymbol{X}_j)$ (for $i \leq m$). Now $\boldsymbol{X}(t)$ is the structure obtained by first taking n_1 copies of the $\boldsymbol{X}_i$ with $\boldsymbol{e}(\boldsymbol{X}_i) = \boldsymbol{w}_1$, then n_2 copies of the $\boldsymbol{X}_i$ with $\boldsymbol{e}(\boldsymbol{X}_i) = \boldsymbol{w}_2$, et cetera, and finally, after taking n_m copies of the $\boldsymbol{X}_i$ with $\boldsymbol{e}(\boldsymbol{X}_i) = \boldsymbol{w}_m$, adding another vertex which is connected with the roots of all the previous trees. Of course once we have defined $\boldsymbol{X}(t)$ we can extract X_t.

It is straightforward to formalize this definition in the logic IFP+C, though it requires some tedious work. ⊣

As a matter of fact, for all classes for which we know that isomorphism coincides with C^k-equivalence for some $k \geq 1$ we also know that they have IFP+C-definable canonization. But this is only an empirical observation, it is not known whether

$$(\cong \restriction_{\mathcal{C}}) = (\equiv_{\mathsf{C}^k} \restriction_{\mathcal{C}})$$

always implies that $\mathcal{C}$ admits IFP+C-definable canonization.

THEOREM 7.7.

(i) (*Grohe* [32]) *The class of planar graphs admits* IFP+C-*definable canonization*.

(ii) (*Grohe, Mariño* [33]) *For each $w \geq 1$, the class of graphs of tree-width at most w admits* IFP+C-*definable canonization*.

The following result is implicit in Otto's [64].

PROPOSITION 7.8. *Let $\mathcal{C}$ be a class of τ-structures and $k \in \mathbb{N}$.*

(i) *If I^k is* PTIME-*invertible on $\mathcal{C}$ then $\equiv_{\mathsf{L}^k}$ on $\mathcal{C}$ admits* IFP-*definable canonization*.

(ii) *If J^k is* PTIME-*invertible on $\mathcal{C}$ then $\equiv_{\mathsf{C}^k}$ on $\mathcal{C}$ admits* IFP+C-*definable canonization*.

Let us only consider statement (ii). The idea of the proof is as follows: It is not hard to see that there is an IFP+C-interpretation of τ_{J^k} in τ that maps each structure $\boldsymbol{A}$ to $J^k(\boldsymbol{A})$. By Theorem 8.5 (of the next section) on ordered structures we have IFP+C = PTIME. Thus once we have defined the ordered structure $J^k(\boldsymbol{A})$, we can simulate the PTIME-inversion in IFP+C.

REMARK 7.9. There is an intermediate statement that also has interesting applications, for example, to the class of all binary trees:

Let $\mathcal{C}$ be a class of τ-structures and $k \in \mathbb{N}$. Suppose that there is a PTIME-algorithm that expects a τ_{I^k}-structure $\boldsymbol{I}$ and an integer n (in unary encoding!) as input and computes a τ-structure $\boldsymbol{A}$ of size at most n such that $I^k(\boldsymbol{A}) = \boldsymbol{I}$ (if there is such an $\boldsymbol{A}$). Then $\equiv_{\mathsf{L}^k}$ on $\mathcal{C}$-admits IFP$^{\#}$-definable canonization.

Theorem 6.19 yields:

COROLLARY 7.10. *Let τ be a vocabulary. Then $\equiv_{\mathsf{L}^2}$ on $\mathcal{F}[\tau]$ admits* IFP-*definable canonization and $\equiv_{\mathsf{C}^2}$ on $\mathcal{F}[\tau]$ admits* IFP+C-*definable canonization*.

§8. Capturing complexity classes. We finally come to the point where we can apply our knowledge on finite variable logics to descriptive complexity theory.

Let us review the basic notions of this area. We want to describe complexity classes by logics. This means that for a given complexity class K we want to find a logic L such that computability in K and definability in L coincide.

We have agreed to consider complexity classes as classes of languages over $\{0, 1\}$. However, now we not only want to talk about computations on classes of words (that is, languages), but also on classes of graphs or arbitrary relational structures.

Example 8.1. A well-known result of Hopcroft and Tarjan [44] says that Planarity of graphs is in PTIME. What this means is that there is a PTIME-algorithm that, given *any* adjacency matrix of a graph, decides whether the graph is planar or not.

In general, with each class $\mathcal{C}$ of τ-structures we associate the language

$$\begin{aligned}\mathcal{L}(\mathcal{C}) &= \{\, \boldsymbol{e}((\boldsymbol{A}, \leq^{\boldsymbol{A}})) \mid \boldsymbol{A} \in \mathcal{C}, \leq^{\boldsymbol{A}} \text{ linear order of } A \,\} \\ &= \bigcup_{\boldsymbol{A} \in \mathcal{C}} \mathcal{L}(\boldsymbol{A}).\end{aligned}$$

More concisely, we may write $\mathcal{L}(\mathcal{C}) = \boldsymbol{e}(\mathcal{C}^{\leq})$. We say that $\mathcal{C}$ is K-*computable*, for a complexity class K, if $\mathcal{L}(\mathcal{C}) \in \mathsf{K}$.

Definition 8.2. A logic L *captures* a complexity class K if for each vocabulary τ and each $\mathcal{C} \subseteq \mathcal{F}[\tau]$, the class $\mathcal{C}$ is L-definable if, and only if, it is K-computable.

The archetypal example is Fagin's Theorem:

Theorem 8.3 (Fagin [21]).
Existential second-order logic Σ^1_1 *captures* NPTIME.

The proof of Fagin's Theorem also shows that second-order logic captures the polynomial hierarchy PH. Moreover, it is known that the extension of second-order logic by the partial fixed-point operator captures PSPACE (see [19]).

No such capturing results are known for PTIME or smaller classes. However, there are numerous capturing results for such classes in a restricted setting where only ordered structures are permitted.

Let us formulate these results in a general framework that will be needed later anyway. We call a class of classes of structures of the same vocabulary a *domain*.[6] Hence each domain is a subclass of $\mathsf{F} = \{\, \mathcal{C} \subseteq \mathcal{F}[\tau] \mid \tau \text{ vocabulary} \,\}$.

[6] That is, two structures within a single element of a domain are supposed to have the same vocabulary, whereas the domain itself may contain classes of different vocabularies.

DEFINITION 8.4. A logic L *captures* a complexity class K *on* a domain D if for each $\mathcal{C} \in \mathsf{D}$, the class $\mathcal{C}$ is L-definable if, and only if, it is K-computable.

For each class $\mathcal{C}$ of structures we define the domain $\mathsf{D}(\mathcal{C}) = \{\mathcal{B} \subseteq \mathcal{C}[\tau] \mid \tau \text{ vocabulary}\}$. If L captures K on $\mathsf{D}(\mathcal{C})$ we usually just say that L captures K on $\mathcal{C}$.

As I have mentioned, the class $\mathcal{O}$ of ordered structures plays a central role. As a matter of fact, in the literature *capturing* is often only defined for ordered structures.

For us, the most important capturing result on ordered structures is the following:

THEOREM 8.5 (Immerman [46], Livchak [56], Vardi [70]).
IFP *captures* PTIME *on the class of ordered structures.*

Other well known results, due to Immerman [48], are that deterministic transitive closure logic captures $\mathsf{LOGSPACE}$ and transitive closure logic captures $\mathsf{NLOGSPACE}$ on ordered structures.

The reason for the importance of an order in this context is that the usual proofs of capturing results proceed by simulating Turing machine computations in the respective logic. However, the input of a Turing machine is always the code of an ordered structure, so to get the simulation started we first need an ordered copy of the structure we want to talk about. Sometimes, if the logic is strong enough, we can cope with this problem. For example, to prove Fagin's Theorem 8.3 we can start a Σ^1_1-sentence simulating a given Turing machine by saying "there exists an order such that ... ". Unfortunately, nothing similar is possible in IFP or $\mathsf{IFP}{+}\mathsf{C}$.

There is a another important type of domain that generalizes those of the form $\mathsf{D}(\mathcal{C})$. Let $\equiv$ be an equivalence relation on a class $\mathcal{C}$ of structures. Then we let

$$\mathsf{D}(\mathcal{C}, \equiv) = \{\, \mathcal{B} \subseteq \mathcal{C}[\tau] \mid \tau \text{ vocabulary}, \mathcal{B} \equiv\text{-closed}^7 \,\}.$$

Since by our general assumption, each class of structures is isomorphism-closed, we have $\mathsf{D}(\mathcal{C}) = \mathsf{D}(\mathcal{C}, \cong\restriction_{\mathcal{C}})$ for all classes $\mathcal{C}$.

If a logic L captures a complexity class K on a domain $\mathsf{D}(\mathcal{C}, \equiv)$ we say that L captures $\equiv$-*invariant* K *on* $\mathcal{C}$.

8.1. Effectively capturing complexity classes. Our notion of a logic capturing a complexity class still needs to be refined to exclude pathological examples such as the following:

EXAMPLE 8.6 (Gurevich [35]). The following logic captures PTIME: Its sentences are all pairs (M, p), where M a Turing machine and p a polynomial. A τ-structure $\boldsymbol{A}$ is a model of a sentence (M, p) if M accepts $\mathcal{L}(\mathcal{C})$ for a class $\mathcal{C}$ of τ-structures that contains $\boldsymbol{A}$ in at most $p(\text{inputlength})$ steps.

[7] $\mathcal{B}$ is $\equiv$-*closed* if for all $\boldsymbol{A}, \boldsymbol{B}$ such that $\boldsymbol{A} \equiv \boldsymbol{B}$ we have $\boldsymbol{A} \in \mathcal{B} \iff \boldsymbol{B} \in \mathcal{B}$.

Classical definitions of a logic, such as the notion of a *regular logic* (see [18]) do not suffice for our purpose; in addition our logics are supposed to satisfy certain effectivity conditions. Gurevich [35] suggested the following definitions.

DEFINITION 8.7. A *logic* is a pair $(L, \models_L)$ where L is function that assigns a recursive set $L[\tau]$ of sentences to each vocabulary τ, and $\models_L$ is a binary relation between sentences and structures such that for all $\varphi \in L[\tau]$ the class

$$\mathrm{Mod}(\varphi) = \{\, \boldsymbol{A} \in \mathcal{F} \mid \boldsymbol{A} \models_L \varphi \,\}$$

is an (isomorphism closed) class of τ-structures.

We say that a class $\mathcal{C}$ of τ-structures is *definable* in $(L, \models_L)$ if there is an $L[\tau]$-sentence φ such that $\mathcal{C} = \mathrm{Mod}(\varphi)$.

Note that the logic in Example 8.6 is a logic in the sense of this definition. For the natural complexity classes K such as

$$\mathsf{PTIME}, \quad \mathsf{NPTIME}, \quad \mathsf{LOGSPACE}, \quad \text{et cetera}$$

we can define the notion of a *clocked* K*-Turing machine*. For example, a clocked PTIME-Turing machine is a pair (M, p) where M is a deterministic Turing machine and p a polynomial such that M runs at most $p(n)$ steps on an input of length n.

DEFINITION 8.8. A logic $\mathsf{L} = (L, \models_L)$ *effectively captures* a complexity class K *on* a domain D if the following two conditions hold:

(i) Each K-computable class in D is L-definable.
(ii) For all vocabularies τ there is a recursive mapping M that associates a clocked K-Turing machine $M(\varphi)$ with each sentence $\varphi \in L[\tau]$ so that if $\mathrm{Mod}(\varphi) \in \mathsf{D}$ then $M(\varphi)$ accepts the language $\mathcal{L}(\mathrm{Mod}(\varphi))$.

We say that L *effectively captures* K if L effectively captures K on F.

Clearly, a logic that effectively captures a complexity class captures the class. All capturing results mentioned in the previous subsection are effective. However, the logic in Example 8.6 does not capture PTIME effectively, since it is not decidable whether a given Turing machine accepts an (isomorphism closed) class of structures (see [35]).

The notion of a logic effectively capturing a complexity class still seems to be far from what we expect a logic to be. However, it has been justified by a nice result of Dawar [13], which says that if there is a logic that effectively captures a complexity class K then there is a logic of the form $\mathsf{FO}(Q_{\mathcal{C}}^{\omega})$ that effectively captures K. Here $\mathsf{FO}(Q_{\mathcal{C}}^{\omega})$ denotes the logic obtained by adjoining the vectorized Lindström-quantifier associated with a class $\mathcal{C}$ of structures to first-order logic. Logics of the form $\mathsf{FO}(Q_{\mathcal{C}}^{\omega})$ are quite natural from a model theoretic point of view. For details and a discussion of this result I refer the reader to [19].

8.2. Is there a logic that effectively captures PTIME? Whereas we have seen that there are logics effectively capturing the common complexity classes above PTIME, for PTIME and the classes below only capturing results on ordered structures are known. The class PTIME, often considered as the class of all "feasible" problems, plays a central role here. In a sense, a logic that effectively captures PTIME can be considered as a programming language that allows to write programs for all feasible queries, and that allows to write them in such a way that they are automatically representation (order) independent. Representation independence is particularly desirable in database theory, where the physical representation of the structure (database) is hidden to the user.

So it is no surprise that the question for a logic that effectively captures PTIME has first occurred in database theory: Chandra and Harel [11] posed it in a slightly different formulation. As we have seen in the previous subsection, Gurevich [35] made the notion of a logic effectively capturing PTIME precise. He conjectured that no logic effectively captures PTIME.

The truth of this conjecture would imply PTIME $\neq$ NPTIME. Note, however, that this does not immediately follow from Fagin's theorem, because even PTIME $=$ NPTIME would not automatically guarantee that Σ^1_1 effectively captures PTIME. (Condition (ii) of Definition 8.8 might be violated.) To prove that the truth of Gurevich's conjecture does imply PTIME $\neq$ NPTIME it is easiest to apply a result going back to Lovász and Gács [57] that there is a complete problem for Σ^1_1 (and thus for NPTIME, even on unordered structures) under first-order reductions.

So both answers to the question if there is a logic effectively capturing PTIME would be interesting: "Yes" would give a nice database query language, and "No" would separate PTIME from NPTIME. In addition, Dawar's [13] result mentioned at the end of the last subsection would give us a lot of information on the structure of the class PTIME if there was a logic effectively capturing it.

Even if we cannot find a logic that captures PTIME, we can try to capture as much of PTIME as possible.

8.3. The approach through canonization. All known results on effectively capturing PTIME on domains D are based on Theorem 8.5 (in the effective form) and the following fundamental (and well-known) observation.

PROPOSITION 8.9. *Let τ be a vocabulary and $\equiv$ an equivalence relation on a class $\mathcal{C}$ of τ-structures. Suppose that $\mathcal{C}$ is* PTIME*-computable and that $\equiv$ admits* PTIME*-canonization. Then there is a logic that effectively captures $\equiv$-invariant* PTIME *on $\mathcal{C}$.*[8]

PROOF. Let γ be a PTIME-canonization function for $\equiv$. We define a logic $(L, \models_L)$ as follows: Since $\mathcal{C}$ is a class of τ-structures, we can let $L[\sigma] = \emptyset$

[8] Recall the definition on page 388, right before Subsection 8.1.

for all vocabularies $\sigma \neq \tau$. We let $L[\tau]$ be the class of IFP-sentences of vocabulary $\tau \cup \{\leq\}$.

Furthermore, we say that $A \models_L \varphi$, for a τ-structure A and a sentence $\varphi \in L[\tau]$, if $A \in \mathcal{C}$ and $\gamma(A) \models \varphi$ in the usual sense.

It is not hard to see that $(L, \models_L)$ effectively captures $\equiv$-invariant PTIME on $\mathcal{C}$. ⊣

Corollary 8.10. *Suppose that isomorphism on the class of graphs admits* PTIME-*canonization. Then there is a logic that effectively captures* PTIME.

Proof. By well-known techniques of encoding arbitrary structures by graphs we can obtain canonization algorithms for structures of arbitrary vocabularies from a canonization algorithm for graphs. ⊣

Corollary 8.11. *Let $\mathcal{C}$ be a class of graphs of bounded genus or bounded valence. Then there is a logic that effectively captures* PTIME *on $\mathcal{C}$.*

Proof. It follows from results of Filotti and Mayer [22] and independently Miller [59] that there is a PTIME-canonization function for isomorphism on graphs of bounded genus. Babai and Luks [4] proved that there is a PTIME-canonization function for isomorphism on graphs of bounded valence. ⊣

8.4. Definable canonization revisited. The disadvantage of the results in the previous subsection is that the logics we obtain are quite unnatural, even though they satisfy Gurevich's conditions. However, just as we are usually not satisfied with knowing that a complexity class has a complete problem, but want to find concrete problems that are complete, we want to show capturing results for concrete logics.

In this section we give such results for IFP+C.

Note first that Theorem 6.9 implies that IFP+C does not capture PTIME (since isomorphism of graphs of bounded valence is in PTIME). But nevertheless the logic IFP+C does capture important parts of PTIME.

As a consequence of Theorem 8.5 we get:

Proposition 8.12. *Let $k \geq 1$ and $\equiv$ an equivalence relation on an* IFP+C-*definable class $\mathcal{C}$ of τ-structures that admits* IFP+C-*definable canonization. Then* IFP+C *effectively captures* PTIME *on* $\mathsf{D}(\mathcal{C}, \equiv)$.

Proof. Let $\mathcal{D} \in \mathsf{D}(\mathcal{C}, \equiv)$ be PTIME-computable. Then $\mathcal{D}^{\leq} \subseteq \mathcal{O}$ is also PTIME-computable, and by Theorem 8.5 there is an IFP-formula ψ of vocabulary $\tau \cup \{\leq\}$ that defines $\mathcal{D}^{\leq}$.

Let Φ be an IFP+C-interpretation of $\tau \cup \{\leq\}$ in τ that defines a canonization function for $\equiv$. It is a well known fact about syntactical interpretations that we can translate statements about the image of a structure back to statements about the original structure. Thus there is an IFP+C-sentence $\psi^{-\Phi}$ such that for all $A \in \mathcal{C}$ we have

$$A \models \psi^{-\Phi} \iff \Phi^{A} \models \psi.$$

Let φ be an $\mathsf{IFP{+}C}$-sentence that defines $\mathcal{C}$. Then $\varphi \wedge \psi^{-\Phi}$ defines $\mathcal{D}$. $\dashv$

Now the capturing results follow:

COROLLARY 8.13.

(i) (*Immerman*, *Lander* [49], *Lindell* [55])
$\mathsf{IFP{+}C}$ *effectively captures* PTIME *on the class of trees.*

(ii) (*Grohe* [32])
$\mathsf{IFP{+}C}$ *effectively captures* PTIME *on the class of planar graphs.*

(iii) (*Grohe, Mariño* [33])
For each $w \geq 1$, $\mathsf{IFP{+}C}$ *effectively captures* PTIME *on the class of graphs of tree-width at most* w.

(iv) (*Immerman*, *Lander* [49], *Hella*, *Kolaitis*, *Luosto* [41])
$\mathsf{IFP{+}C}$ *effectively captures* PTIME *almost everywhere.* (*More precisely, for each vocabulary* τ *there is a class* $\mathcal{C}$ *of* τ*-structures with* $\mu(\mathcal{C}) = 1$ *such that* $\mathsf{IFP{+}C}$ *effectively captures* PTIME *on* $\mathcal{C}$.)

PROOF. To be able to apply Proposition 8.12, it remains to show that the respective classes are $\mathsf{IFP{+}C}$-definable. This is obvious for the class of trees.

It is proved in [32] that the class of planar graphs is IFP-definable and in [33] that for each $w \geq 1$ the class of graphs of tree-width at most w is IFP-definable.

(iv) follows from Theorem 7.4. $\dashv$

COROLLARY 8.14 (Otto [64]). IFP *effectively captures* $\equiv_{\mathsf{L}^2}$*-invariant* PTIME *and* $\mathsf{IFP{+}C}$ *effectively captures* $\equiv_{\mathsf{C}^2}$*-invariant* PTIME.

PROOF. We fix a vocabulary τ. By Theorem 6.19, the invariant J^2_τ is PTIME-invertible. Thus, by Proposition 7.8, $\equiv_{\mathsf{C}^2}$ on $\mathcal{F}[\tau]$ admits $\mathsf{IFP{+}C}$-definable canonization. By Proposition 8.12, this implies the statement. $\dashv$

§9. Conclusions and further research.

9.1. Model theory of finite variable logics. As we have seen in Sections 3 and 4, finite variable logics have a very nice "basic finite model theory". Its development in the beginning of the 90s created a certain optimism that it may go on in such a smooth way, but since then mostly negative results have been established. In particular, the translation between Diophantine equations and L^k-theories gives us a general construction method for very complicated complete L^k-theories.

It is probably too naive to hope for nice model theoretic results holding on all finite structures. Instead, it seems reasonable to concentrate on particular, well-behaved classes of structures. For example, the theories of trees or planar graphs are good candidates. Of course this requires criteria of "well-behavedness". Model theorists may wonder if there is a concept like "stability" for finite variable theories of finite structures. But the criteria for well-behavedness we should look for in this context, with its close connections to finite combinatorics and computer science, may also be quite different.

A technical aspect that may help to shed some light on these problems is to find a reasonable theory of definability with parameters in finite variable logics.

It would also be interesting to study the finite variable logics on arbitrary, not just finite structures. The gap between finite and infinite structures does not seem as big for the finite variable logics as it is for first-order logic.

The finite variable logics with counting seem to be less natural from a model-theoretic point of view, but they are certainly worthwhile to study. In particular, their close connection to combinatorial questions that we have touched upon in Subsection 2.1 should be further explored. I tend to believe that the logics with counting are even more important for finite model theory than their counterparts without counting.

9.2. Descriptive complexity theory. Though extending the domain on which we can capture PTIME is an obvious and reasonable approach to the question of whether there exists a logic effectively capturing PTIME, not too much work on this question has been done so far. We have mentioned almost all known results in this survey.

Natural candidates for further results are classes satisfying certain topological constraints, such as being embeddeable in surfaces of a fixed genus, or being closed under minors. I conjecture that for each nontrivial class $\mathcal{C}$ of graphs that is closed under minors there is a logic that effectively captures PTIME on $\mathcal{C}$. This may even be true for the logic IFP+C.

On the other hand, we know from Theorem 6.9 that the logic IFP+C has its limits. So we should also look for stronger logics. In particular, it is desirable to find a "nice" logic that can simulate the group theoretic graph-canonization algorithms and therefore capture PTIME, for example, on all classes of graphs of bounded valence. We know that such logics exist, so the emphasis here is on "nice".

It is conceivable that such a logic can again be embedded into suitable finite variable logics, which may then provide the model theoretic framework to handle it. The extension IFP+C of IFP and the corresponding finite variable logics C^k give a perfect example of how useful this can be.

9.3. Technical questions. We close the article with some technical problems.

Recall the definitions and results of Section 5. The vocabulary τ for which we have seen that the invariant I_τ^k has no recursive inverse depends on k—it has to contain at least one $(k-1)$-ary relation symbol. It would be desirable to prove the result for a vocabulary not depending on k, ideally for $\{E\}$.

Open Question 9.1. *Is it true that for each* $k \geq 3$ *there is no recursive inverse*

$$f : \mathcal{O}[\{E\}_{I^k}] \to \mathcal{F}[\{E\}]$$

for $I^k_{\{E\}}$?

Recall that Corollary 5.8 answers a question of Dawar [15] asking for a "finite downward Löwenheim-Skolem theorem" for the L^k. He also asked for a "finite upward Löwenheim-Skolem theorem":

Open Question 9.2. *Let* $k \geq 3$. *Is there a recursive function g such that if a complete* L^k*-theory has a finite model of size larger than g of its k-size, then it has arbitrarily large finite models*?

Note that, as for the downward version, the answer is "yes" if we omit the word "recursive". Also note that our translations relate this question to the following: *Is there a recursive function f such that if a Diophantine equation d has a positive integer solution of size* $f(d)$, *then it has arbitrarily large positive integer solutions?* A negative answer to this question would yield a negative answer to Question 9.2 at least for $k = 3$.

Some of the questions we have studied here can also be asked for arbitrary, not just finite structures. For example, the invariant I^k can be extended to arbitrary structures in a straightforward manner. Many infinite structures, for example $(\mathbb{Q}, \leq)$, have a finite invariant. We may ask:

Open Question 9.3. *Let* $k \geq 3$ *and* τ *a vocabulary that contains at least one binary relation symbol. Is it decidable if a given ordered finite* τ_{I^k}*-structure is* $I^k(\boldsymbol{A})$ *for an arbitrary* τ*-structure* $\boldsymbol{A}$?

I have already mentioned that for the counting logics the main questions concerning inversion of the invariant and canonization are still open. For the sake of completeness, let me pose them here again:

Open Question 9.4. *Let* $k \geq 3$ *and* τ *a vocabulary that contains at least one binary relation symbol. Is there a* PTIME*-computable inverse*

$$f : \mathcal{O}[\tau_{J^k}] \to \mathcal{F}[\tau]$$

of J^k_τ?

Open Question 9.5. *Let* $k \geq 3$ *and* τ *a vocabulary that contains at least one binary relation symbol. Is there a* PTIME*-computable canonization function for* C^k*-equivalence on the class of all finite* τ*-structures*?

Our results on definable canonization and Proposition 6.25 suggest the following question:

Open Question 9.6. *Let* $k \geq 1$ *and* $\mathcal{C}$ *a class of structures such that isomorphism coincides with* C^k*-equivalence on*

$$\mathcal{C}^{\preceq} = \{\, (\boldsymbol{A}, \preceq) \mid \boldsymbol{A} \in \mathcal{C}, \preceq \text{ is a pre-order on } A \,\}.$$

Does $\mathcal{C}$ *admit* IFP+C*-definable canonization*?

In general, the relation between isomorphism testing, canonization, and capturing polynomial time demands further clarification.

Acknowledgements. I would like to thank Andreas Blass, Jörg Flum, Markus Frick, Lauri Hella, Markus Junker, Julian Mariño, Martin Otto, and Thomas Schwentick for valuable comments on earlier versions of this article.

REFERENCES

[1] S. Abiteboul, M. Y. Vardi, and V. Vianu, *Fixpoint logics, relational machines, and computational complexity*, ***Journal of the ACM***, vol. 44 (1997), pp. 30–56.

[2] S. Abiteboul and V. Vianu, *Generic computation and its complexity*, ***Proceedings of the 23rd ACM symposium on theory of computing***, 1991, pp. 209–219.

[3] L. Babai, P. Erdös, and S. Selkow, *Random graph isomorphism*, ***SIAM Journal on Computing***, vol. 9 (1980), pp. 628–635.

[4] L. Babai and E. M. Luks, *Canonical labeling of graphs*, ***Proceedings of the 15th ACM symposium on theory of computing***, 1983, pp. 171–183.

[5] J. Barwise, *Back and forth through infinitary logic*, ***Studies in model theory*** (M. D. Morley, editor), Studies in Mathematics, Mathematical Association of America, 1973, pp. 5–34.

[6] ———, *On Moschovakis closure ordinals*, this Journal, vol. 42 (1977), pp. 292–296.

[7] A. Blass, Y. Gurevich, and D. Kozen, *A zero-one law for logic with a fixed-point operator*, ***Information and Control***, vol. 67 (1985), pp. 70–90.

[8] E. Börger, E. Grädel, and Y. Gurevich, ***The classical decision problem***, Springer-Verlag, 1997.

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

[10] P. Cameron, ***Combinatorics: Topics, techniques, algorithms***, Cambridge University Press, 1994.

[11] A. Chandra and D. Harel, *Structure and complexity of relational queries*, ***Journal of Computer and System Sciences***, vol. 25 (1982), pp. 99–128.

[12] A. Dawar, *Feasible computation through model theory*, ***Ph.D. thesis***, University of Pennsylvania, 1993.

[13] ———, *Generalized quantifiers and logical reducibilities*, ***Journal of Logic and Computation***, vol. 5 (1995), pp. 213–226.

[14] ———, *A restricted second order logic for finite structures*, ***Logic and computational complexity: International workshop, LCC '94*** (D. Leivant, editor), Lecture Notes in Computer Science, vol. 960, Springer-Verlag, 1995.

[15] ———, *Types and indiscernibles in finite models*, ***Logic colloquium '95*** (J. A. Makowsky and E. V. Ravve, editors), Lecture Notes in Logic, vol. 11, Springer-Verlag, 1998, pp. 51–65.

[16] A. Dawar, L. Hella, and Ph. G. Kolaitis, *Implicit definability and infinitary logic in finite model theory*, ***Proceedings of the 22nd international colloquium on automata, languages and programming***, 1995, pp. 624–635.

[17] A. Dawar, S. Lindell, and S. Weinstein, *Infinitary logic and inductive definability over finite structures*, ***Information and Computation***, vol. 119 (1995), pp. 160–175.

[18] H.-D. Ebbinghaus, *Extended logics: The general framework*, ***Model-theoretic logics*** (J. Barwise and S. Feferman, editors), Springer-Verlag, 1985, pp. 25–76.

[19] H.-D. Ebbinghaus and J. Flum, ***Finite model theory***, Springer-Verlag, 1995.

[20] H.-D. Ebbinghaus, J. Flum, and W. Thomas, ***Mathematical logic***, 2nd ed., Springer-Verlag, 1994.

[21] R. Fagin, *Generalized first-order spectra and polynomial-time recognizable sets*, ***Complexity of computation*** (R. M. Karp, editor), SIAM-AMS Proceedings, vol. 7, 1974, pp. 43–73.

[22] I. S. Filotti and J. N. Mayer, *A polynomial-time algorithm for determining the isomorphism of graphs of fixed genus*, ***Proceedings of the 12th ACM symposium on theory of computing***, 1980, pp. 236–243.

[23] J. Flum and M. Ziegler, private communication.

[24] E. Grädel, Ph. G. Kolaitis, and M. Y. Vardi, *On the decision problem for two-variable first-order logic*, ***Bulletin of Symbolic Logic***, vol. 3 (1997), pp. 53–69.

[25] E. Grädel and M. Otto, *Inductive definability with counting on finite structures*, ***Computer science logic, 6th workshop, CSL '92, San Miniato 1992, selected papers*** (E. Börger, G. Jäger, H. Kleine Büning, S. Martini, and M. M. Richter, editors), Lecture Notes in Computer Science, vol. 702, Springer-Verlag, 1993, pp. 231–247.

[26] E. Grädel and M. Otto, *On logics with two variables*, ***Theoretical Computer Science***, to appear, 1998.

[27] E. Grädel, M. Otto, and E. Rosen, *Two-variable logic with counting is decidable*, ***Proceedings of the 12th IEEE symposium on logic in computer science***, 1997, pp. 306–317.

[28] ———, *Undecidability results on two-variable logics*, ***Proceedings of the 14th annual symposium on theoretical aspects of computer science***, 1997, pp. 249–260.

[29] M. Grohe, *Equivalence in finite-variable logics is complete for polynomial time*, ***Proceedings of the 37th annual IEEE symposium on foundations of computer science***, 1996, pp. 264–273.

[30] ———, *Large finite structures with few* L^k*-types*, ***Proceedings of the 12th IEEE symposium on logic in computer science***, 1997, pp. 216–227.

[31] ———, *Canonization for* L^k*-equivalence is hard*, ***Computer science logic, 11th international workshop CSL '97 (Selected papers)*** (M. Nielsen and W. Thomas, editors), Lecture Notes in Computer Science, vol. 1414, Springer-Verlag, 1998.

[32] ———, *Fixed-point logics on planar graphs*, ***Proceedings of the 13th IEEE symposium on logic in computer science***, 1998, pp. 6–15.

[33] M. Grohe and J. Mariño, *Definability and descriptive complexity on databases of bounded tree-width*, ***Proceedings of the 7th international conference on database theory*** (C. Beeri, editor), Lecture Notes in Computer Science, Springer-Verlag, 1999, to appear.

[34] Y. Gurevich, *Toward logic tailored for computational complexity*, ***Computation and proof theory*** (M. M. Richter, E. Börger, W. Oberschelp, B. Schinzel, and W. Thomas, editors), Lecture Notes in Mathematics, vol. 1104, Springer-Verlag, 1984, pp. 175–216.

[35] ———, *Logic and the challenge of computer science*, ***Current trends in theoretical computer science*** (E. Börger, editor), Computer Science Press, 1988, pp. 1–57.

[36] ———, *From invariants to canonization*, ***Bulletin of the European Association for Theoretical Computer Science***, vol. 63 (1997), pp. 115–119.

[37] Y. Gurevich and S. Shelah, *Fixed point extensions of first-order logic*, ***Annals of Pure and Applied Logic***, vol. 32 (1986), pp. 265–280.

[38] ———, *On finite rigid structures*, this Journal, vol. 61 (1996), pp. 549–562.

[39] L. Hella, *Definability hierarchies of generalized quantifiers*, ***Annals of Pure and Applied Logic***, vol. 43 (1989), pp. 235–271.

[40] ———, *Logical hierarchies in* PTIME, ***Information and Computation***, vol. 129 (1996), pp. 1–19.

[41] L. Hella, Ph. G. Kolaitis, and K. Luosto, *Almost everywhere equivalence of logics in finite model theory*, ***Bulletin of Symbolic Logic***, vol. 2 (1996), pp. 422–443.

[42] ———, *How to define a linear order on finite models*, ***Annals of Pure and Applied Logic***, vol. 87 (1997), pp. 241–267.

[43] W. Hodges, ***Model theory***, Cambridge University Press, 1993.

[44] J. E. Hopcroft and R. Tarjan, *Efficient planarity testing*, ***Journal of the ACM***, vol. ? (1974), pp. 549–568.

[45] N. Immerman, *Upper and lower bounds for first-order expressibility*, ***Journal of Computer and System Sciences***, vol. 25 (1982), pp. 76–98.

[46] ———, *Relational queries computable in polynomial time*, ***Information and Control***, vol. 68 (1986), pp. 86–104.

[47] ———, *Expressibility as a complexity measure: results and directions*, ***Proceedings of the 2nd IEEE symposium on structure in complexity theory***, 1987, pp. 194–202.

[48] ———, *Languages that capture complexity classes*, ***SIAM Journal on Computing***, vol. 16 (1987), pp. 760–778.

[49] N. Immerman and E. Lander, *Describing graphs: A first-order approach to graph canonization*, ***Complexity theory retrospective*** (A. Selman, editor), Springer-Verlag, 1990, pp. 59–81.

[50] R. M. Karp and R. J. Lipton, *Turing machines that take advice*, ***L'Enseignement Mathématique***, vol. 28 (1982), pp. 191–209.

[51] Ph. G. Kolaitis and M. Y. Vardi, *Fixpoint logic vs. infinitary logic in finite-model theory*, ***Proceedings of the 7th IEEE symposium on logic in computer science***, 1992, pp. 46–57.

[52] ———, *Infinitary logic and* 0-1 *laws*, ***Information and Computation***, vol. 98 (1992), pp. 258–294.

[53] ———, *Infinitary logic for computer science*, ***Proceedings of the 19th international colloquium on automata, languages and programming***, Lecture Notes in Computer Science, vol. 623, Springer-Verlag, 1992, pp. 450–473.

[54] ———, *On the expressive power of variable-confined logics*, ***Proceedings of the 11th IEEE symposium on logic in computer science***, 1996.

[55] S. Lindell, *An analysis of fixed-point queries on binary trees*, ***Theoretical Computer Science***, vol. 85 (1991), pp. 75–95.

[56] A. Livchak, *The relational model for process control*, ***Automated Documentation and Mathematical Linguistics***, vol. 4 (1983), pp. 27–29.

[57] L. Lovász and P. Gács, *Some remarks on generalized spectra*, ***Zeitschrift für mathematische Logik und Grundlagen der Mathematik***, vol. 23 (1977), pp. 27–144.

[58] J. V. Matijasevič, *Enumerable sets are diophantic*, ***Soviet Mathematics Doklady***, vol. 11 (1970), pp. 345–357.

[59] G. L. Miller, *Isomorphism testing for graphs of bounded genus*, ***Proceedings of the 12th ACM symposium on theory of computing***, 1980, pp. 225–235.

[60] M. Mortimer, *On languages with two variables*, ***Zeitschrift für mathematische Logik und Grundlagen der Mathematik***, vol. ? (1975), pp. 135–140.

[61] Y. N. Moschovakis, *On nonmonotone inductive definability*, ***Fundamenta Mathematicae***, vol. 82 (1974), pp. 39–83.

[62] M. Otto, *Ptime canonization for two variables with counting*, ***Proceedings of the 10th IEEE symposium on logic in computer science***, 1995, pp. 342–352.

[63] ———, *The expressive power of fixed-point logic with counting*, this Journal, vol. 61 (1996), pp. 147–176.

[64] ———, ***Bounded variable logics and counting***, Lecture Notes in Logic, vol. 9, Springer-Verlag, 1997.

[65] L. Pacholski, W. Szwast, and L. Tendera, *Complexity of two-variable logic with counting*, ***Proceedings of the 12th IEEE symposium on logic in computer science***, 1997, pp. 318–327.

[66] C. H. Papadimitriou, ***Computational complexity***, Addison-Wesley, 1994.

[67] B. Poizat, *Deux ou trois choses que je sais de L_n*, this Journal, vol. 47 (1982), pp. 641–658.

[68] D. SCOTT, *Logics with denumerably long formulas and finite strings of quantifiers*, ***Theory of models*** (J. W. Addison, L. Henkin, and A. Tarski, editors), North-Holland, 1965, pp. 329–341.

[69] A. SETH, *When do fixed point logics capture complexity classes?*, ***Proceedings of the 10th IEEE symposium on logic in computer science***, 1995, pp. 353–363.

[70] M. Y. VARDI, *The complexity of relational query languages*, ***Proceedings of the 14th ACM symposium on theory of computing***, 1982, pp. 137–146.

[71] ———, *On the complexity of bounded-variable queries*, ***Proceedings of the 14th ACM symposium on principles of database systems***, 1995, pp. 266–276.

INSTITUT FÜR MATHEMATISCHE LOGIK
ECKERSTRASSE 1
79104 FREIBURG, GERMANY
E-mail: grohe@sun2.mathematik.uni-freiburg.de