On the Expressive Power of Logics on Finite Models

Phokion G. Kolaitis*
Computer Science Department
University of California, Santa Cruz
Santa Cruz, CA 95064, USA
kolaitis@cs.ucsc.edu

August 1, 2003

*Partially supported by NSF Grant IIS-9907419

Contents

1 Introduction **3**

2 Basic Concepts **3**

3 Ehrenfeucht-Fraïssé Games for First-Order Logic **9**

4 Computational Complexity **22**
4.1 Complexity Classes 22
4.2 The Complexity of Logic 23

5 Ehrenfeucht-Fraïssé Games for Existential Second-Order Logic **26**

6 Logics with Fixed-Point Operators **30**
6.1 Operators and Fixed-Points 30
6.2 Least Fixed-Point Logic 33
6.3 Datalog and Datalog($\neq$) 42
6.4 The Complementation Problem for LFP_1 and a Normal Form for LFP 46
6.5 Partial Fixed-Point Logic 49

7 Infinitary Logics with Finitely Many Variables **52**
7.1 The Infinitary Logic $\mathrm{L}^{\omega}_{\infty\omega}$ 52
7.2 Pebble Games and $\mathrm{L}^{\omega}_{\infty\omega}$-Definability 54
7.3 0-1 Laws for $\mathrm{L}^{\omega}_{\infty\omega}$ 61
7.4 Definability and Complexity of $\mathrm{L}^{k}_{\infty\omega}$-Equivalence 63
7.5 Least Fixed-Point Logic vs. Partial Fixed-Point Logic on Finite Structures 67

8 Existential Infinitary Logics with Finitely Many Variables **69**
8.1 The infinitary logics $\exists\mathrm{L}^{k}_{\infty\omega}$ and $\exists\mathrm{L}^{k}_{\infty\omega}(\neq)$ 69
8.2 Existential Pebble Games 71
8.3 Descriptive Complexity of Fixed Subgraph Homeomorphism Queries 77

9 References **79**

1 Introduction

Finite model theory can be succintly described as the study of logics on classes of finite structures. In addition to first-order logic, various other logics have been explored in the context of finite model theory, including fragments of second-order logic, logics with fixed-point operators, infinitary logics, and logics with generalized quantifiers. Typical classes of finite structures on which these logics have been investigated are the class of all finite graphs, the class of all finite ordered graphs, the class of all finite planar graphs, the class of all finite strings, and the class of all finite trees.

Finite model theory provides a conceptual and methodological framework for exploring the connections between logic and several key areas of computer science, such as database theory, computational complexity, and computer-aided verification. This is perhaps the primary motivation for developing finite model theory. As its development progressed, however, it became clear that finite model theory is an area of research that deserves to be studied in its own right. While the traditional focus of mathematical logic has been on fixed infinite structures or on classes of finite and infinite structures, it turned out that new phenomena emerge, when one focuses on classes of finite structures. These phenomena give finite model theory its own distinctive character and set it apart from other areas of mathematical logic.

There are three main areas of reseach in finite model theory: the study of the expressive power of logics on finite structures; the study of the connections between logic and computational complexity, an area which is also known as descriptive complexity; and the study of the connections between logic and asymptotic probabilities. The first of these three areas is the focus of the present chapter.

2 Basic Concepts

A *vocabulary* is a finite set $\sigma = \{R_1, \ldots, R_m, c_1, \ldots, c_s\}$ of relation symbols of specified arities and constant symbols. A *σ-structure* is a tuple $\mathbf{A} = (A, R_1^{\mathbf{A}}, \ldots, R_m^{\mathbf{A}}, c_1^{\mathbf{A}}, \ldots, c_s^{\mathbf{A}})$ such that A is a a non-empty set, called the *universe* of $\mathbf{A}$, each $R_i^{\mathbf{A}}$ is a relations on A such that $\text{arity}(R_i^{\mathbf{A}}) = \text{arity}(R_i)$, $1 \leq i \leq m$, and each $c_j^{\mathbf{A}}$ is a distinguished element of A, $1 \leq j \leq s$. A *finite σ-structure* is a σ-structure $\mathbf{A}$ whose universe A is a finite set. If the vocabulary is understood from the context, we simply use the terms "structure" and "finite structure". Also, whenever no confusion arises and in order to simplify the notation, we will use the same symbol for both a relation (constant) symbol and the relation (distinguished element) interpreting it on a structure.

Let $\mathbf{A} = (A, R_1^{\mathbf{A}}, \ldots, R_m^{\mathbf{A}}, c_1^{\mathbf{A}}, \ldots, c_s^{\mathbf{A}})$ and $\mathbf{B} = (B, R_1^{\mathbf{B}}, \ldots, R_m^{\mathbf{B}}, c_1^{\mathbf{B}}, \ldots, c_s^{\mathbf{B}})$ be two σ-structures. An *isomorphism* between $\mathbf{A}$ to $\mathbf{B}$ is a mapping $h : A \to B$ that satisfies the following conditions:

- h is a one-to-one and onto function.
- For every constant symbol c_j, $1 \leq j \leq s$, we have that $h(c_j^{\mathbf{A}}) = c_j^{\mathbf{B}}$.
- For every relation symbol R_i, $1 \leq i \leq m$, of arity t and for every t-tuple $(a_1, \ldots, a_t)$ from A, we have that $R_i^{\mathbf{A}}(a_1, \ldots, a_t)$ if and only if $R_i^{\mathbf{B}}(h(a_1), \ldots, h(a_t))$.

A structure $\mathbf{B} = (B, R_1^{\mathbf{B}}, \ldots, R_m^{\mathbf{B}}, c_1^{\mathbf{B}}, \ldots, c_s^{\mathbf{B}})$ is a *substructure* of $\mathbf{A}$ if $B \subseteq A$, each $R_i^{\mathbf{B}}$ is the restriction of $R_i^{\mathbf{B}}$ to B (which means that $R_i^{\mathbf{B}} = R_i^{\mathbf{A}} \cap B^t$, where t is the arity of R_i), $1 \leq i \leq m$, and $c_j^{\mathbf{B}} = c_j^{\mathbf{A}}$, $1 \leq j \leq s$. If $\mathbf{A}$ is a σ-structure and D is a subset of A, then the *substructure of $\mathbf{A}$ generated by D* is the structure $\mathbf{A} \restriction D$ having the set $D \cup \{c_1^{\mathbf{A}}, \ldots, c_s^{\mathbf{A}}\}$ as its universe and having the restrictions of the relations $R_i^{\mathbf{A}}$ on $D \cup \{c_1^{\mathbf{A}}, \ldots, c_s^{\mathbf{A}}\}$ as its relations.

A *partial isomorphism from* $\mathbf{A}$ *to* $\mathbf{B}$ is an isomorphism from a substructure of $\mathbf{A}$ to a substructure of $\mathbf{B}$. From the preceding definitions, it follows that every partial isomorphism from $\mathbf{A}$ to $\mathbf{B}$ must map each constant $c_j^{\mathbf{A}}$ of $\mathbf{A}$ to the constant $c_j^{\mathbf{B}}$, $1 \leq j \leq s$.

The following examples illustrate some of these concepts. A *graph* is a structure $\mathbf{G} = (V, E)$, where E is a binary relation on V. The subgraph of $\mathbf{G}$ induced by a set D of nodes is precisely the substructure of $\mathbf{G}$ generated by D. A *graph with two distinguished nodes* s *and* t is a structure $\mathbf{G} = (V, E, s, t)$. An *ordered graph* is a structure $\mathbf{G} = (V, E, \leq)$, where E is a binary relation on V and $\leq$ is a linear order on V. A k*-colored graph* is a structure $\mathbf{G} = (V, E, P_1, \ldots, P_k)$, where E is a binary relation on V and each P_i is a unary relation on V consisting of all nodes of color i, $1 \leq i \leq k$. Finally, a *binary string* of length n can be thought of as a structure $\mathbf{S} = (\{1, 2, \ldots, n\}, P)$, where P is a unary relation on $\{1, \ldots, n\}$ such that $i \in P$ if and only if the i-th bit of the string is equal to 1, where $1 \leq i \leq n$. For instance, the string 10001 can be identified with the finite structure $(\{1, 2, 3, 4, 5\}, \{1, 5\})$.

The concept of a *query*, which originated in database theory, is one of the most fundamental concepts in finite model theory. We now give the precise definition and present several examples.

Definition 2.1: Let σ be a vocabulary and k a positive integer.

- A *class of* σ*-structures* is a collection $\mathcal{C}$ of σ-structures that is *closed under isomorphisms*, which means that if $\mathbf{A} \in \mathcal{C}$ and $\mathbf{B}$ is a structure that is isomorphic to $\mathbf{A}$, then $\mathbf{B} \in \mathcal{C}$.
- A k*-ary query on a class* $\mathcal{C}$ is a mapping Q with domain $\mathcal{C}$ and such that
  - $Q(\mathbf{A})$ is a k-ary relation on $\mathbf{A}$, for $\mathbf{A} \in \mathcal{C}$;
  - Q is *preserved under isomorphisms*, which means that if $h : \mathbf{A} \to \mathbf{B}$ is an isomorphism, then $Q(\mathbf{B}) = h(Q(\mathbf{A}))$.
- A *Boolean query on a class* $\mathcal{C}$ is a mapping $Q : \mathcal{C} \to \{0, 1\}$ that is preserved under isomorphisms i.e., if $\mathbf{A}$ is isomorphic to $\mathbf{B}$, then $Q(\mathbf{A}) = Q(\mathbf{B})$. Consequently, Q can be identified with the subclass $\mathcal{C}' = \{\mathbf{A} \in \mathcal{C} : Q(\mathbf{A}) = 1\}$ of $\mathcal{C}$. ∎

Example 2.2: Consider the following queries on graphs $\mathbf{G} = (V, E)$.

- The Transitive Closure query TC is the binary query such that

$$TC(\mathbf{G}) = \{(a, b) \in V^2 : \text{there is a path from } a \text{ to } b\}.$$

- The 2-Disjoint Paths query is the 4-ary query $2DP$ such that

$$2DP(\mathbf{G}) = \{(a, b, c, d) \in V^4 : \text{there are two node-disjoint paths path from } a \text{ to } b \text{ and from } c \text{ to } d\}.$$

- The Articulation Point query is the unary query AP such that

$$AP(\mathbf{G}) = \{a \in V : a \text{ is an articulation point of } \mathbf{G}\}.$$

- The Even Cardinality query $EVEN$ is the Boolean query such that

$$EVEN(\mathbf{G}) = \begin{cases} 1 & \text{if } \mathbf{G} \text{ has an even number of nodes} \\ 0 & \text{otherwise.} \end{cases}$$

- The CONNECTIVITY query CN is the Boolean query such that

$$CN(\mathbf{G}) \;=\; \begin{cases} 1 & \text{if } \mathbf{G} \text{ is connected} \\ 0 & \text{otherwise.} \end{cases}$$

- The Boolean queries EULERIAN, ACYCLICITY, k-COLORABILITY, and HAMILTONIAN PATH are defined in an analogous way.

∎

Queries are mathematical objects that formalize the concept of a "property" of structures and elements of structures. This formalization makes it possible to define and study what it means for such a "property" to be expressible in some logic. In other words, we will use logic as a *specification language* of "properties" of structures and elements of structures.

Definition 2.3: Let L be a logic and $\mathcal{C}$ a class of σ-structures

- A k-ary query Q on $\mathcal{C}$ is *L-definable* if there is an L-formula $\varphi(x_1, \ldots, x_k)$ with $x_1, \ldots, x_k$ as free variables and such that for every $\mathbf{A} \in \mathcal{C}$

$$Q(\mathbf{A}) \;=\; \{(a_1, \ldots, a_k) \in A^k : \mathbf{A} \models \varphi(a_1, \ldots, a_k)\}.$$

- A Boolean query Q on $\mathcal{C}$ is *L-definable* if there is an L-sentence ψ such that for every $\mathbf{A} \in \mathcal{C}$

$$Q(\mathbf{A}) = 1 \iff \mathbf{A} \models \psi.$$

- $L(\mathcal{C})$ denotes the collection of all L-definable queries on $\mathcal{C}$. ∎

Two remarks are in order now. First, it should be emphasized that the concept of an L-definable query Q on a class $\mathcal{C}$ of σ-structures is a *uniform definability* concept. This means the same L-formula serves as a specification of the query on every structure in $\mathcal{C}$, which is entirely analogous to the requirement that an algorithm for a problem must produce the correct answer on every instance of the problem. Along these lines, note that if a query Q is L-definable on $\mathcal{C}$ and $\mathcal{C}'$ is a subclass of $\mathcal{C}$, then the restriction of Q on $\mathcal{C}'$ is also L-definable using the formula that defines it on $\mathcal{C}$. Second, the concept of an L-definable query on a class $\mathcal{C}$ makes sense for an arbitrary class of σ-structures that may very well consist of both finite and infinite structures, or only infinite structures, or only finite structures. In particular, this concept contains the following important cases as special cases:

1. $\mathcal{C}$ is the class $\mathcal{S}$ of all (finite and infinite) σ-structures.

   This is the primary case of uniform definability studied in classical model theory.

2. $\mathcal{C}$ consists of a single infinite structure $\mathbf{A}$ (and all its isomorphic copies).

   This is the case of *local definability* on a fixed structure. The two primary examples are the structure $\mathbf{N} = (N, +, \times)$ of arithmetic and the structure $\mathbf{R} = (R, +, \times)$ of analysis, where N is the set of all natural numbers and R is the set of all real numbers.

3. $\mathcal{C}$ is the class $\mathcal{F}$ of finite σ-structures.

   This is the primary case of uniform definability studied in finite model theory.

We now present several examples of queries that are definable in first-order logic or in fragments of second-order logic. We assume familiarity with the syntax and the semantics of first-order logic and second-order logic (see [End72] for the precise definitions). Informally, first-order logic FO over a vocabulary σ has (first-order) variables that are interpreted by elements of the structure at hand, has atomic formulas of the form $s_1 = s_2$ and $R_i(s_1, \ldots, s_t)$, where R_i is a relation symbol and each s_j is a variable or a constant symbol, has the standard propositional connectives $\neg$, $\vee$, $\wedge$, $\rightarrow$, and, finally, has first-order quantifiers $\forall x$ and $\exists x$, for each variable x, that range over elements of the universe of the structure at hand.

Example 2.4: The following queries are first-order definable on the class of all (finite or infinite) graphs.

- The Boolean query "the graph **G** has an isolated node" is definable by the first-order formula

$$(\exists x)(\forall y)(\neg E(x, y) \wedge \neg E(y, x)).$$

- The unary query "the node x has at least two distinct neighbors" is definable by the first-order formula:

$$(\exists y)(\exists z)(\neg(y = z) \wedge E(x, y) \wedge E(x, z)).$$

  Similarly, for each fixed k, the Boolean query "**G** is a k-regular graph" (i.e., each node has exactly k neighbors) is first-order definable.

- The binary query "there is a path of length 3 from x to y" is definable by the first-order formula

$$(\exists z_1)(\exists z_2)(E(x, z_1) \wedge E(z_1, z_2) \wedge E(z_2, y)).$$

∎

The syntax of second-order logic SO is obtained by augmenting the syntax of first-order logic with second-order variables $X, Y, \ldots$ and second-order quantifiers $\exists X, \exists Y, \ldots, \forall X, \forall Y, \ldots$ that are interpreted by relations of fixed arities over the universe of the structure at hand. *Existential second-order logic* ESO and *universal second-order logic* USO are the syntactically simplest fragments of second-order logic. Specifically, ESO consists of all second-order formulas of the form

$$(\exists S_1) \cdots (\exists S_m)\varphi(\overline{x}, S_1, \ldots, S_m),$$

where each S_i is a second-order variable, $1 \leq i \leq m$, and $\varphi(\overline{x}, S_1, \ldots, S_m)$ is a first-order formula. In a dual manner, USO consists of all second-order formulas of the form

$$(\forall S_1) \cdots (\forall S_m)\varphi(\overline{x}, S_1, \ldots, S_m),$$

where each S_i is a second-order variable, $1 \leq i \leq m$, and $\varphi(\overline{x}, S_1, \ldots, S_m)$ is a first-order formula. *Monadic Second-Order Logic* MSO is the fragment of second-order logic consisting of all second-order formulas in which every second-order quantifier is applied to a unary second-order variable, which means that all second-order quantifiers in the formula range over subsets of the universes of structures. *Existential monadic second-order logic* consists of all formulas that are both ESO-formulas and monadic second-order formulas. Similarly, *universal monadic second-order logic* consists of all formulas that are both USO-formulas and monadic second-order formulas.

Example 2.5: The following queries are definable in existential monadic second-order logic on the class of all (finite or infinite) graphs:

1. The Boolean query DISCONNECTIVITY is definable by the formula

$$(\exists S)((\exists x)S(x) \wedge (\exists y)\neg S(y) \wedge (\forall z)(\forall w)(S(z) \wedge \neg S(w) \rightarrow \neg E(z,w))).$$

   Intutively, this sentence asserts that there are two disjoint, non-empty sets of nodes with no edge between them.

2. The Boolean query 2-COLORABILITY is definable by the formula

$$(\exists R)(\forall x)(\forall y)(E(x,y) \rightarrow (R(x) \leftrightarrow \neg R(y))).$$

   Intutively, the two colors are encoded by R and the complement of R.

3. For every $k \geq 3$, the Boolean query k-COLORABILITY is definable by a formula of existential monadic second-order logic with $k-1$ existential monadic quantifiers. The formula is similar to the one above used to define 2-COLORABILITY: each of the $k-1$ monadic second order variables encodes a different color, while the k-th color is encoded by the complement of the union of these $k-1$ colors.

∎

Example 2.6: The WELL-FOUNDEDNESS Boolean query is definable on the class of all linear orders $(V, \leq)$ by the following formula of universal monadic second-order logic:

$$(\forall S)((\exists x)S(x) \rightarrow (\exists y)(S(y) \wedge (\forall z)(S(z) \rightarrow y \leq z))).$$

∎

Example 2.7: The Boolean query HAMILTONIAN PATH is definable on the class of all finite graphs $G = (V, E)$ by an existential second-order formula that asserts that

$$(\exists T)((\text{“}T \text{ is a linear order on } V\text{”}) \wedge (\forall x)(\forall y)(\text{“}y \text{ is the successor of } x \text{ in } T\text{”} \rightarrow E(x,y))),$$

where T is a second-order variable of arity 2. In the above formula, the properties “T is a linear order on V” and “y is the successor of x in T” are clearly expressible in first-order logic. ∎

Example 2.8: The Boolean query RIGIDITY (i.e., given a graph $\mathbf{G} = (V, E)$, is the identity function its only automorphism?) is definable on the class of all finite graphs by a universal second-order formula that asserts that

$$(\forall S)(\text{“}S \text{ encodes an automorphism of } \mathbf{G}\text{”} \rightarrow (\forall x)S(x,x)),$$

where S is a binary relation symbol. ∎

The expressive power of a logic L on a class $\mathcal{C}$ of finite structures is measured by the collection $L(\mathcal{C})$ of L-definable queries on $\mathcal{C}$. As a general rule, the expressive power of a logic L is *context-dependent*, that it to say, $L(\mathcal{C})$ depends on the class $\mathcal{C}$ on which the logic L is studied. For instance, first-order logic has very high expressive power on the structure $\mathbf{N} = (N, +, \times)$ of arithmetic, since every recursively enumerable relation is first-order definable on $\mathbf{N}$. In contrast, first-order logic

has limited expressive power on the class of all (finite or infinite) graphs, since properties as basic as CONNECTIVITY and ACYCLICITY are not first-order definable. First-order logic has limited expressive power on the class of all finite graphs as well. In particular, none of the following queries is first-order definable on finite graphs: EVEN CARDINALITY, CONNECTIVITY, ACYCLICITY, PLANARITY, EULERIAN, k-COLORABILITY, for each fixed $k \geq 2$, and HAMILTONIAN PATH. Actually, it is fair to say that *no* property of finite graphs that requires recursion is first-order definable.

The central question about the expressive power of a logic L on a class $\mathcal{C}$ of structures is to determine which queries on $\mathcal{C}$ are L-definable and which are not. Clearly, to show that a query Q on $\mathcal{C}$ is L-definable, it suffices to find some L-formula that defines it on every structure in $\mathcal{C}$. In contrast, showing that Q is *not* L-definable is in principle a more challenging task, since it entails showing that *no* formula of L defines the property. In many respects, this is analogous to the difference between establishing upper and lower bounds on the computational complexity of an algorithmic problem. For this reason, much of the investigation of the expressive power of a logic centers on the development of techniques for showing that queries are not definable in that logic.

There are three main tools for investigating the expressive power of first-order logic:

- The *Compactness Theorem*;
- The method of *Ultraproducts*;
- The method of *Ehrenfeucht-Fraïssé Games*.

The compactness theorem and the method of ultraproducts are direct and effective tools for analyzing the expressive power of first-order logic on the class of all (finite or infinite) structures over a given vocabulary. To illustrate this point, let us recall the standard proof that CONNECTIVITY is not first-order definable on the class of all graphs. Towards a contradiction, assume that there is a first-order sentence ψ such that for every graph $\mathbf{G} = (V, E)$ we have that $\mathbf{G} \models \psi$ if and only if $\mathbf{G}$ is connected. Let c', d' be two constant symbols and, for every $n \geq 1$, let φ_n be a first-order sentence asserting that there is no path of length n from c to d. Then every finite subset of the set

$$T = \{\varphi_n : n \geq 1\} \cup \{\psi\}$$

has a model (for instance, a sufficiently long path with c and d as its endpoints). Consequently, the compactness theorem implies that T has a model $\mathbf{G} = (V, E, c, d)$. This, however, gives rise to a contradiction. Indeed, on the one hand $\mathbf{G}$ is connected, since $\mathbf{G} \models \psi$; in particular, there is a path from (the distinguished element interpreting) c to (the distinguished element interpreting) d in $\mathbf{G}$. On the other hand, however, there is no path from c to d in $\mathbf{G}$, since $\mathbf{G} \models \varphi_{\mathbf{n}}$, for every $n \geq 1$.

Although the above proof establishes that CONNECTIVITY is not first-order definable on the class of all graphs, it does *not* establish that this property is not first-order definable on the class of all finite graphs. The reason is that the model of T guaranteed to exist by the compactness theorem need not be finite. In general, it may very well be the case that every finite subset of a set T of first-order sentences has a finite model, but T itself has only infinite models. Therefore, a proof that uses the compactness theorem to show that a query is not first-order definable on all structures does not automatically translate to a proof that the query is not first-order definable on all finite structures. Similar obstacles arise when using the method of ultraproducts. While it is still possible to use the compactness theorem and the methods of ultraproducts to study the expressive power of first-order logic on finite structures [GV85], the use of these tools is often somewhat cumbersome or not intuitive. In contrast, the method of Ehrenfeucht-Fraïssé games is a tool that has been successfully applied to the study of first-order logic in finite model theory.

Furthermore, it is a flexible and extendible tool, since variants of Ehrenfeucht-Fraïssé games can be formulated and used to study the expressive power of logics that are stronger than first-order logic and do not possess the compactness theorem.

3 Ehrenfeucht-Fraïssé Games for First-Order Logic

This section is devoted to a presentation of the Ehrenfeucht-Fraïssé games and their applications to the analysis of the expressive power of first-order logic on finite structures.

Definition 3.1: Let r be a positive integer, σ a vocabulary, and $\mathbf{A}$ and $\mathbf{B}$ two σ-structures.

The *r-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$* is played between two players, called the *Spoiler* and the *Duplicator*, according to the following rules:

Each run of the game has r moves. In each move, the Spoiler plays first and picks an element from the universe A of $\mathbf{A}$ or from the universe B of $\mathbf{B}$; the Duplicator then responds by picking an element from the universe of the other structure (i.e., if the Spoiler has picked an element from A, then the Duplicator picks an element from B, and vice versa). Let $a_i \in A$ and $b_i \in B$ be the two elements picked by the Spoiler and the Duplicator in their i-th move, $1 \leq i \leq r$.

- The *Duplicator wins the run* $(a_1, b_1), \ldots, (a_r, b_r)$ if the mapping

  $$a_i \mapsto b_i,\ 1 \leq i \leq r, \text{ and } c_i^{\mathbf{A}} \mapsto c_j^{\mathbf{B}},\ 1 \leq j \leq s,$$

  is a partial isomorphism from $\mathbf{A}$ to $\mathbf{B}$, which means that it is an isomorphism between the substructure $\mathbf{A} \restriction \{a_1, \ldots, a_r\}$ of $\mathbf{A}$ generated by $\{a_1, \ldots, a_r\}$ and the substructure $\mathbf{B} \restriction \{b_1, \ldots, b_r\}$ of $\mathbf{B}$ generated by $\{b_1, \ldots, b_r\}$.

  Otherwise, *the Spoiler wins the run* $(a_1, b_1), \ldots, (a_r, b_r)$.

- The *Duplicator wins the r-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$* if the Duplicator can win every run of the game, i.e., if (s)he has a *winning strategy* for the Ehrenfeucht-Fraïssé game.

  Otherwise, the *Spoiler wins the r-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$*.

- We write $\mathbf{A} \sim_r \mathbf{B}$ to denote that the Duplicator wins the r-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$. ∎

Spoiler	$a_1 \in A$	$b_2 \in B$	$b_3 \in B$	$\ldots$	$a_r \in A$
	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\ldots$	$\updownarrow$
Duplicator	$b_1 \in B$	$a_2 \in A$	$a_3 \in A$	$\ldots$	$b_r \in B$

Figure 1: A typical run of the r-move Ehrenfeucht-Fraïssé-game

The next proposition follows immediately from Definition 3.1.

Proposition 3.2: *$\sim_r$ is an equivalence relation on the class $\mathcal{S}$ of all σ-structures.*

Example 3.3: Let $\mathbf{A}$ and $\mathbf{B}$ be the graphs depicted in Figure 2. Then

- $\mathbf{A} \sim_2 \mathbf{B}$, i.e., the Duplicator wins the 2-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$, $\mathbf{B}$;

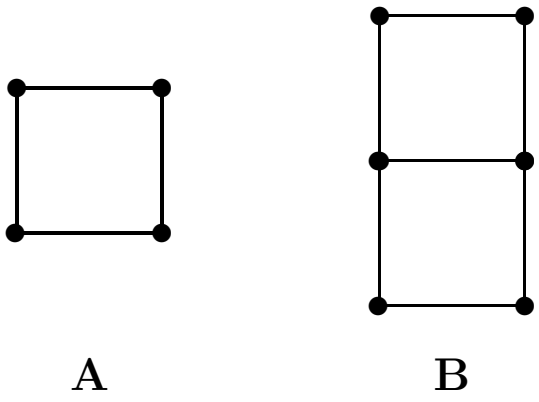


Figure 2: A difference between the 2-move and the 3-move Ehrenfeucht-Fraïssé game

- $\mathbf{A} \not\sim_3 \mathbf{B}$, i.e., the Spoiler wins the 3-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$, $\mathbf{B}$.

The Duplicator can win the 2-move game by playing in such a way that there is an edge between a_1 and a_2 if and only if there is an edge between b_1 and b_2. In contrast, the Spoiler can win the 3-move game by picking three elements in $\mathbf{B}$ with no edge between any two of them. ■

Note that the description of a winning strategy for the Duplicator in the Ehrenfeucht-Fraïssé-game, as presented in Definition 3.1, is rather informal. The concept of a winning strategy for the Duplicator can be made precise, however, in terms of families of partial isomorphisms with appropriate extension properties.

Definition 3.4: A *winning strategy for the Duplicator* in the r-move Ehrenfeucht-Fraïssé-game on $\mathbf{A}$ and $\mathbf{B}$ is a sequence $I_0, I_1, \ldots, I_r$ of non-empty sets of partial isomorphims from $\mathbf{A}$ to $\mathbf{B}$ such that

- The sequence $I_0, I_1, \ldots, I_r$ has the *forth property*:

  For every $i < r$, every $f \in I_i$, and every $a \in A$, there is a $g \in I_{i+1}$ such that $a \in \mathrm{dom}(g)$ and $f \subseteq g$.

- The sequence $I_0, I_1, \ldots, I_r$ has the *back property*:

  For every $i < r$, every $f \in I_i$, and every $b \in B$, there is a $g \in I_{i+1}$ such that $b \in \mathrm{rng}(g)$ and $f \subseteq g$.

In effect, the forth property provides the Duplicator with a good move when the Spoiler picks an element of $\mathbf{A}$, while the back property provides the Duplicator with a good move when the Spoiler picks an element of $\mathbf{B}$.

The key feature of Ehrenfeucht-Fraïssé games is that they capture the combinatorial content of first-order quantification; for this reason, Ehrenfeucht-Fraïssé games can be used to characterize definability in first-order logic on an arbitrary class of σ-structures. To describe the precise connection between first-order logic and Ehrenfeucht-Fraïssé games, we need to bring into the picture a well-known concept from mathematical logic.

Definition 3.5: Let φ be a first-order formula over a vocabulary σ. The *quantifier rank* of φ, denoted by $\mathrm{qr}(\varphi)$, is the depth of quantifier nesting in φ. More formally, $\mathrm{qr}(\varphi)$ is defined by the following induction on the construction of φ:

- If φ is atomic, then $\mathrm{qr}(\varphi) = 0$.
- If φ is of the form $\neg\psi$, then $\mathrm{qr}(\varphi) = \mathrm{qr}(\psi)$.
- If φ is of the form $\psi_1 \wedge \psi_2$ or of the form $\psi_1 \vee \psi_2$, then $\mathrm{qr}(\varphi) = \max\{\mathrm{qr}(\psi_1), \mathrm{qr}(\psi_2)\}$.
- If φ is of the form $\exists x\psi$ or of the form $\forall x\psi$, then $\mathrm{qr}(\varphi) = \mathrm{qr}(\psi) + 1$. ∎

Note that if a first-order formula is in prenex normal form, then its quantifier rank is equal to the number of the quantifiers in its prefix. For instance, if φ is $(\forall x)(\forall y)(\exists z)\theta$, where θ is quantifier-free, then $\mathrm{qr}(\varphi) = 3$. In contrast, if φ is $(\exists x)E(x,x) \vee (\exists y)(\forall z)\neg E(y,z)$, then $\mathrm{qr}(\varphi) = 2$. Note also that if $\mathrm{qr}(\varphi) = r$, then for every $r' > r$ there is a first-order formula ψ such that $\mathrm{qr}(\psi) = r'$ and φ is logically equivalent to ψ.

Definition 3.6: Let r be a positive integer, and let $\mathbf{A}$ and $\mathbf{B}$ be two σ-structures. We write $\mathbf{A} \equiv_r \mathbf{B}$ to denote that $\mathbf{A}$ and $\mathbf{B}$ satisfy the same first-order sentences of quantifier rank r. ∎

Proposition 3.7: *$\equiv_r$ is an equivalence relation on the class $\mathcal{S}$ of all σ-structures.*

Note that the equivalence relation $\equiv_r$ is defined using purely logical concepts. The main technical result of this section asserts that $\equiv_r$ coincides with the equivalence relation $\sim_r$, which was defined using purely combinatorial concepts.

Theorem 3.8: ([Fra54, Ehr61]) *Let r be a positive integer, and let $\mathbf{A}$ and $\mathbf{B}$ be two σ-structures. Then the following statements are equivalent:*

1. *$\mathbf{A} \equiv_r \mathbf{B}$, i.e., $\mathbf{A}$ and $\mathbf{B}$ satisfy the same first-order sentences of quantifier rank r.*
2. *$\mathbf{A} \sim_r \mathbf{B}$, i.e., the Duplicator wins the r-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$.*

Moreover, the following are true:

- *$\equiv_r$ has finitely many equivalence classes.*
- *Each $\equiv_r$-equivalence class is definable by a first-order sentence of quantifier rank r.*

Example 3.9: Before embarking on the proof of Theorem 3.8, let us briefly revisit Example 3.3. As seen in that example, the Spoiler wins the 3-move Ehrenfeucht-Fraïssé game on the structures $\mathbf{A}$ and $\mathbf{B}$ in Figure 2. Therefore, Theorem 3.8 tells that there is first-order sentence of quantifier rank 3 that is satisfied by one of the two structures, but not by the other. Indeed, if φ is the sentence

$$\exists x\exists y\exists z(x \neq y \wedge x \neq z \wedge y \neq z \wedge \neg E(x,y) \wedge \neg E(x,z) \wedge \neg E(y,z)),$$

then $\mathbf{B} \models \varphi$, but $\mathbf{A} \not\models \varphi$. Note also that this sentence yields a strategy for the Spoiler to win the 3-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$: the Spoiler picks three elements b_1, b_2, b_3 from B such that $\mathbf{B}, b_1, b_2, b_3 \models (x \neq y \wedge x \neq z \wedge y \neq z \wedge \neg E(x,y) \wedge \neg E(x,z) \wedge \neg E(y,z))$. Another sentence witnessing that $\mathbf{A} \not\equiv_3 \mathbf{B}$ is the sentence

$$(\forall x)(\forall y)(\exists z)(x \neq y \rightarrow E(x,z) \wedge E(y,z)),$$

which is true on $\mathbf{A}$, but is false on $\mathbf{B}$. In turn, this sentence yields another strategy for the Spoiler to win the 3-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$:

The Spoiler first picks two elements from B such that $\mathbf{B}, b_1, b_2 \models \forall z\neg(x \neq y \rightarrow E(x,z) \wedge E(y,z))$. After the Duplicator has picked elements a_1, a_2 from A, the Spoiler picks an element a_3 from A such that $\mathbf{A}, a_1, a_2, a_3 \models x \neq y \rightarrow E(x,z) \wedge E(y,z)$; the Duplicator is unable to respond to this move in a way that a partial isomorphism is maintained. ∎

$$\begin{array}{lccc} \text{Spoiler} & b_1 \in B & b_2 \in B & a_3 \in A \\ & \updownarrow & \updownarrow & \updownarrow \\ \text{Duplicator} & a_1 \in A & a_2 \in A & b_3 \in B. \end{array}$$

We now proceed with the proof of Theorem 3.8. One part of it is has a relatively straightforward proof.

Theorem 3.10: *Let r be a positive integer. If $\mathbf{A}$ and $\mathbf{B}$ are two σ-structures such that the Duplicator wins the Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$, then every first-order sentence of quantifier rank r that is true on $\mathbf{A}$ is also true on $\mathbf{B}$. Consequently, if $\mathbf{A} \sim_r \mathbf{B}$, then $\mathbf{A} \equiv_r \mathbf{B}$.*

Proof: We proceed by induction on the quantifier rank of formulas. Assume that the result holds for all formulas of quantifier rank r over an arbitrary vocabulary. We have to show that if φ is a formula of quantifier rank $r+1$ and $\mathbf{A}$, $\mathbf{B}$ are two σ-structures such that $\mathbf{A} \sim_{r+1} \mathbf{B}$ and $\mathbf{A} \models \varphi$, then $\mathbf{B} \models \varphi$. The interesting cases are the ones in which φ is of the form $\exists x\psi$ or of the form $\forall x\psi$.

Assume that φ is of the form $\exists x\psi$, which implies that $\mathrm{qr}(\psi) = r$. We have to show that $\mathbf{B} \models \exists x\psi$. Since $\mathbf{A} \models \varphi$, there is an element $a \in A$ such that $\mathbf{A}, a \models \psi$. Let c be a new constant symbol and let $\psi[x/c]$ be the first-order sentence obtained from ψ by replacing every free occurrence of the variable x by c. Clearly, $\psi[x/c]$ is a sentence of quantifier rank σ over the vocabulary $\sigma \cup \{c\}$. Now view the above element $a \in A$ as the first move of the Spoiler in a run the $(r+1)$-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$. Let $b \in B$ be the response of the Duplicator in this game played according to the Duplicator's winning strategy. Therefore, the Duplicator wins the r-move Ehrenfeucht-Fraïssé game on $(\mathbf{A}, a)$ and $(\mathbf{B}, b)$ viewed as structures over the vocabulary $\sigma \cup \{c\}$. Moreover, $(\mathbf{A}, a) \models \psi[x/c]$, so the induction hypothesis implies that $(\mathbf{B}, b) \models \psi[x/c])$, which, in turn, implies that $\mathbf{B} \models \exists x\psi$.

Next, assume that φ is of the form $\forall x\psi$, which again implies that $\mathrm{qr}(\psi) = r$. We have to show that $\mathbf{B} \models \forall x\psi$. Let b be an arbitrary element of B. View this element as the first move of the Spoiler in a run of the $(r+1)$-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$. Let $a \in A$ be the response of the Duplicator in this game played according to the Duplicator's winning strategy. Therefore, the Duplicator wins the r-move Ehrenfeucht-Fraïssé game on $(\mathbf{A}, a)$ and $(\mathbf{B}, b)$ viewed as structures over the vocabulary $\sigma \cup \{c\}$, where, as in the previous case, c is a new constant symbol. Since $\mathbf{A} \models \forall x\psi$, we have that $(\mathbf{A}, a) \models \psi[x/c]$. Consequently, the induction hypothesis implies that $(\mathbf{B}, b) \models \psi[x/c])$. ∎

To prove the remaining parts of Theorem 3.8, we need to first introduce the concept of an (m, r)-*type*, $0 \leq m \leq r$, and establish some basic properties of this concept. The definition of an (m, r)-type is by backward induction on m.

Definition 3.11: Assume that σ is a vocabulary, r is a positive integer, and $x_1, \ldots, x_r$ are variables of first-order logic.

- An (r, r)-*type* is a conjunction of atomic or negated atomic formulas over the vocabulary σ such that every variable occurring in this conjunction is one of the variables $x_1, \ldots, x_r$ and for every atomic formula θ over σ with variables among $x_1, \ldots, x_r$ either θ or $\neg\theta$ occurs as a conjunct.

- Assume that the concept of $(m+1,r)$-type has been defined, where $0 \leq m \leq r-1$. An (m,r)-*type* is an expression of the form

  $$\bigwedge\{\exists x_{m+1}\varphi : \varphi \text{ is an } (m+1,r)\text{-type in } S\} \wedge \bigwedge\{\forall x_{m+1}\neg\varphi : \varphi \text{ is an } (m+1,r)\text{-type not in } S\},$$

  where S is a subset of the set of all $(m+1,r)$-types. ∎

Lemma 3.12: *Let σ be a vocabulary, r a positive integer, and m an integer such that $0 \leq m \leq r$.*

- *Every (m,r)-type is a first-order formula over the vocabulary σ such that its free variables are among $x_1, \ldots, x_m$ and its quantifier rank is $r-m$.*
- *There are only finitely many distinct (m,r)-types.*
- *For every σ-structure $\mathbf{A}$ and every sequence $a_1, \ldots, a_m$ of elements of A, there is exactly one (m,r)-type φ such that $\mathbf{A}, a_1, \ldots, a_m \models \varphi$.*

Proof: We use backward induction on m. Since σ consists of finitely many relation and constant symbols, there are finitely many atomic and negated atomic formulas over σ with variables among $x_1, \ldots, x_r$. It follows that every (r,r)-type is a finite conjunction of such formulas and, thus, it is a first-order formula of quantifier rank 0. Moreover, every sequence $a_1, \ldots, a_m$ of elements from the universe of a structure $\mathbf{A}$ satisfies a unique (r,r)-type, namely the conjunction of all atomic and negated atomic formulas over σ that are satisfied by this tuple.

Assume that the properties of the lemma hold for $(m+1,r)$-types. In particular, the set of all $(m+1,r)$-types is finite, hence it has finitely many subsets, which implies that there are finitely many (m,r)-types. Moreover, the defining expression of an (m,r)-type is a first-order formula of quantifier rank $r-m$, since each $(m+1,r)$-type is a first-order formula of quantifier rank $r-(m+1) = r-m-1$. Finally, assume that $\mathbf{A}$ is a σ-structure and $a_1, \ldots, a_m$ is a sequence of elements from A. Let S^* be the set of all $(m+1,r)$-types φ such that $\mathbf{A}, a_1, \ldots, a_m \models \exists x_{m+1}\varphi$. Then $\mathbf{A}, a_1, \ldots, a_m$ satisfies the (m,r)-type determined by S^*, i.e., the formula

$$\bigwedge\{\exists x_{m+1}\varphi : \varphi \in S^*\} \wedge \bigwedge\{\forall x_{m+1}\neg\varphi : \varphi \notin S^*\},$$

where φ ranges over all (m,r)-types. Moreover, if $\mathbf{A}, a_1, \ldots, a_m$ satisfies some other (m,r)-type determined by a set S, then it is easy to see that $S = S^*$, so $\mathbf{A}, a_1, \ldots, a_m$ satisfies a unique (m,r)-type. ∎

Definition 3.13: Let σ be a vocabulary, r a positive integer, m an integer such that $0 \leq m \leq r$, $\mathbf{A}$ a σ-structure and $a_1, \ldots, a_m$ a sequence of elements from the universe of $\mathbf{A}$.

We write $\varphi_r^{\mathbf{A},a_1,\ldots,a_r}$ to denote the unique (m,r)-type satisfied by $\mathbf{A}, a_1, \ldots, a_m$. In particular, when $m = 0$, we write $\varphi_r^{\mathbf{A}}$ to denote the unique $(0,r)$-type satisfied by $\mathbf{A}$. ∎

According to Lemma 3.12, each expression $\varphi_r^{\mathbf{A},a_1,\ldots,a_m}$ is a first-order formula of quantifier rank $r-m$ with free variables among $x_1, \ldots, x_m$. In particular, each $\varphi_r^{\mathbf{A}}$ is a first-order sentence of quantifier rank r. It should be pointed out that the assumption that the vocabulary σ consists of finitely many relation and constant symbols was critical in showing that each (m,r)-type is a first-order formula and also that, for each m and each r with $0 \leq r \leq m$, there are finitely many distinct (m,r)-types. We are now ready to complete the proof of Theorem 3.8 and also establish that $\varphi_r^{\mathbf{A}}$ defines the $\equiv_r$-equivalence class of $\mathbf{A}$.

Theorem 3.14: *Let r be a positive integer, let $\mathbf{A}$ and $\mathbf{B}$ be two σ-structures, and let $\varphi_r^{\mathbf{A}}$ be the unique $(0,r)$-type satisfied by $\mathbf{A}$. Then the following statements are equivalent:*

1. $\mathbf{A} \equiv_r \mathbf{B}$, *i.e., $\mathbf{A}$ and $\mathbf{B}$ satisfy the same first-order sentences of quantifier rank r.*
2. $\mathbf{B} \models \varphi_r^{\mathbf{A}}$.
3. $\mathbf{A} \sim_r \mathbf{B}$, *i.e., the Duplicator wins the r-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$.*

Proof: The implication (1) $\Rightarrow$ (2) follows from the definitions and the fact that $\varphi_r^{\mathbf{A}}$ is satisfied by $\mathbf{A}$ and has quantifier-rank r. The implication (3) $\Rightarrow$ (1) was established in Theorem 3.10. Consequently, it remains to prove the implication (2) $\Rightarrow$ (3).

Assume that $\mathbf{B} \models \varphi_r^{\mathbf{A}}$. We describe a winning strategy for the Duplicator in the r-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$. The key property of the Duplicator's strategy is that, for every run of the game and for every integer m with $0 \leq m \leq r$, if $a_1, \ldots, a_m$ and $b_1, \ldots, b_m$ are the elements of $\mathbf{A}$ and $\mathbf{B}$ played in the first m-moves of that run, then $\mathbf{A}, a_1, \ldots, a_m$ and $\mathbf{B}, b_1, \ldots, b_m$ satisfy the same (m,r)-type.

Assume first that the Spoiler begins by playing an element a_1 from A. Let $\varphi_r^{\mathbf{A},a_1}$ be the unique $(1,r)$-type satisfied by $\mathbf{A}, a_1$. Hence, the sentence $\exists x_1 \varphi_r^{\mathbf{A},a_1}$ is a conjunct of $\varphi_r^{\mathbf{A}}$, which implies that $\mathbf{B} \models \exists x_1 \varphi_r^{\mathbf{A},a_1}$. Let b_1 be an element of B such that $\mathbf{B}, b_1 \models \varphi_r^{\mathbf{A},a_1}$. This element b_1 is the Duplicator's response to the Spoiler's first move. Assume then that the Spoiler begins by playing an element b_1 from B. Let $\varphi_r^{\mathbf{B},b_1}$ be the unique $(1,r)$-type satisfied by $\mathbf{B}, b_1$. We claim that $\mathbf{A} \models \exists x_1 \varphi_r^{\mathbf{B},b_1}$. Indeed, otherwise, $\mathbf{A} \models \forall x_1 \neg \varphi_r^{\mathbf{B},b_1}$, which implies that $\forall x_1 \neg \varphi_r^{\mathbf{B},b_1}$ is a conjunct of $\varphi_r^{\mathbf{A}}$. Consequently, $\mathbf{B} \models \forall x_1 \neg \varphi_r^{\mathbf{B},b_1}$, which contradicts the fact that $\mathbf{B}, b_1 \models \varphi_r^{\mathbf{B},b_1}$.

By continuing to play this way, the Duplicator ensures that at the end of the run the sequences $a_1, \ldots, a_r$ and $b_1, \ldots, b_m$ are such that $\mathbf{A}, a_1, \ldots, a_r$ and $\mathbf{A}, b_1, \ldots, b_r$ satisfy the same (r,r)-type, i.e., the same atomic and negated atomic formulas. This implies that the mapping $a_i \mapsto b_i$, $1 \leq i \leq r$, is a partial isomorphism. ∎

The first application of the preceding results is a characterization of first-order definability on arbitrary classes of structures.

Theorem 3.15: *Let σ be a vocabulary, $\mathcal{C}$ a class of σ-structures, and Q a Boolean query on $\mathcal{C}$. Then the following statements are equivalent:*

1. *Q is first-order definable on $\mathcal{C}$.*
2. *There is a positive integer r such that, for every structure $\mathbf{A} \in \mathcal{C}$ and every structure $\mathbf{B} \in \mathcal{C}$, if $Q(\mathbf{A}) = 1$ and the Duplicator wins the r-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$, then $Q(\mathbf{B}) = 1$.*

Proof: The implication (1) $\Rightarrow$ (2) is an immediate consequence of Theorem 3.10. For the other direction, assume that such a positive integer r exists. Let S be the set of all $(0,r)$-types of structures $\mathbf{A}$ in $\mathcal{C}$ such that $Q(\mathbf{A}) = 1$. Lemma 3.12 implies that S is a finite set, hence the disjunction

$$\bigvee \{\varphi_r^{\mathbf{A}} : \mathbf{A} \in \mathcal{C} \text{ and } Q(\mathbf{A}) = 1\}$$

is a first-order sentence, which we denote by φ. We now claim that φ defines the query Q on $\mathcal{C}$. Indeed, if $\mathbf{B}$ is a structure in $\mathcal{C}$ such that $Q(\mathbf{B}) = 1$, then its $(0,r)$-type $\varphi_r^{\mathbf{B}}$ is one of the disjuncts of φ, so $\mathbf{B} \models \varphi$. Conversely, if $\mathbf{B}$ is a structure in $\mathcal{C}$ such that $\mathbf{B} \models \varphi$, then there is a structure $\mathbf{A}$

in $\mathcal{C}$ such that $Q(\mathbf{A}) = 1$ and $\mathbf{B} \models \varphi_r^{\mathbf{A}}$. Theorem 3.14 implies that the Duplicator wins the r-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$, hence $Q(\mathbf{B}) = 1$. ∎

The preceding Theorem 3.15 gives rise to a combinatorial method for studying first-order definability and obtaining lower bounds on the expressive power of first-order logic on arbitrary classes of structures.

Method 3.16: The Method of Ehrenfeucht-Fraïssé Games for FO
Let σ be a vocaculary, $\mathcal{C}$ a class of σ-structures, and Q a Boolean query on $\mathcal{C}$.

Soundness: To show that Q is *not* first-order definable on $\mathcal{C}$, it suffices to show that for every positive integer r there are structures $\mathbf{A}_r$ and $\mathbf{B}_r$ in $\mathcal{C}$ such that

- $Q(\mathbf{A}_r) = 1$ and $Q(\mathbf{B}_r) = 0$.
- The Duplicator wins the r-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$.

Completeness: This method is also *complete*, that is, if Q is *not* first-order definable on $\mathcal{C}$, then for every positive integer r such structures $\mathbf{A}_r$ and $\mathbf{B}_r$ exist. ∎

Note that the soundness of the method of Ehrenfeucht-Fraïssé games follows from Theorem 3.10, which is the easier part in establishing that the two equivalence relations $\sim_r$ and $\equiv_r$ coincide. In contrast, the completeness of the method requires Theorem 3.14. We now illustrate this method with two easy applications.

Proposition 3.17: *The* Even Cardinality *query is not first-order definable on the class of all finite graphs.*

Proof: For every $n \geq 1$, let $\overline{\mathbf{K}}_n$ be the totally disconnected graph with n nodes. It is obvious

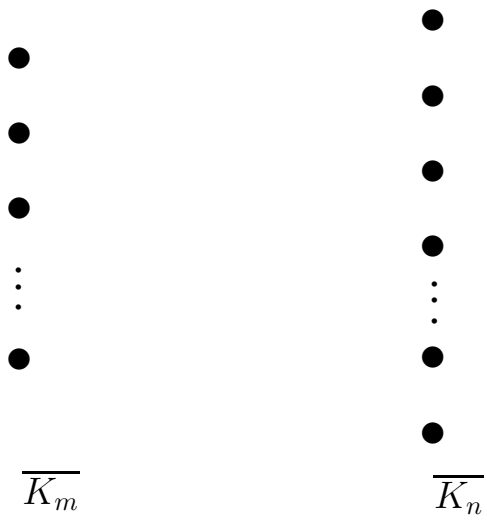


Figure 3: Even Cardinality is not first-order definable on finite graphs

that, for every $r \geq 1$, every $m \geq r$, and every $n \geq r$, the Duplicator wins the r-move Ehrenfeucht-Fraïssé game on $\overline{\mathbf{K}}_m$ and $\overline{\mathbf{K}}_n$. Thus, we can apply the method of Ehrenfeucht-Fraïssé games using the structures $\overline{\mathbf{K}}_m$ with $m \geq r$ an even number and $\overline{\mathbf{K}}_n$ with $n \geq r$ an odd number. ∎

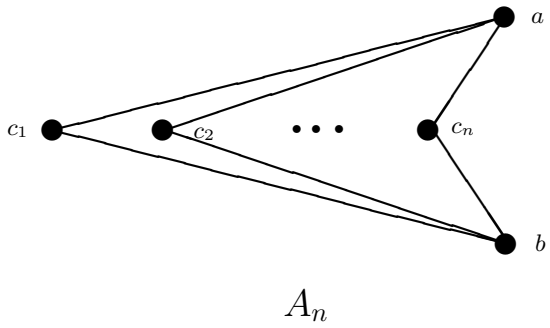


Figure 4: EULERIAN is not first-order definable on finite graphs

Proposition 3.18: *The* EULERIAN *query is not first-order definable on the class of all finite graphs.*

Proof: By definition, a graph is Eulerian if there is a closed walk that traverses each edge exactly once. Euler showed that this property holds if and only if every node has even degree, i.e., an even number of neighbors. For every $n \geq 1$, let $\mathbf{A}_n$ be the graph depicted in Figure 4. Clearly, $\mathbf{A}_n$ is Eulerian if and only if n is an even number. Moreover, for every $n \geq r$, the Duplicator wins the r-move Ehrenfeucht-Fraïssé game on $\mathbf{A}_n$ and $\mathbf{A}_{n+1}$. Thus, we can apply the method of Ehrenfeucht-Fraïssé games using the structures $\mathbf{A}_{2n}$ and $\mathbf{A}_{2n+1}$ with $2n \geq r$. ∎

As seen earlier, the method of Ehrenfeucht-Fraïssé games is complete, which implies that if a query Q is not first-order definable on a class $\mathcal{C}$ of structures, then in principle this can be established using the method of Ehrenfeucht-Fraïssé games. In practice, however, the following technical difficulties may arise when one attempts to apply this method to concrete queries:

- How does one find, for every $r \geq 1$, structures $\mathbf{A}_r$ and $\mathbf{B}_r$ in $\mathcal{C}$ such that $Q(\mathbf{A}_r) = 1$, $Q(\mathbf{B}_r) = 0$, and the Duplicator wins the r-move Ehrenfeucht-Fraïssé game on $\mathbf{A}_r$ and $\mathbf{B}_r$?
- After such candidate structures $\mathbf{A}_r$ and $\mathbf{B}_r$ have been identified, how does one show rigorously that $\mathbf{A}_r \sim_r \mathbf{B}_r$?

As a general rule, both these tasks can be challenging. Nonetheless, they can be eased by pursuing the following two approaches.

- Whenever possible, analyze the $\sim_r$-equivalence classes, $r \geq 1$, of the structures in $\mathcal{C}$ and obtain explicit descriptions of them.
- Find general *sufficient* conditions for the Duplicator to win the r-move Ehrenfeucht-Fraïssé game and, thus, build a "library" of winning strategies for the Duplicator in this game.

The class $\mathcal{L}$ of all finite linear orders provides an interesting, albeit rather rare, case in which it is possible to analyze the $\sim_r$-equivalence classes, $r \geq 1$. Before presenting the full analysis, we give a motivating example. For every $n \geq 1$, we let $\mathbf{L}_n = (\{1, \ldots, n\}, \leq)$ be the standard linear order on $\{1, \ldots, n\}$.

Example 3.19: The following are true for the 3-move Ehrenfeucht-Fraïssé game.

- The Spoiler wins the 3-move Ehrenfeucht-Fraïssé game on $\mathbf{L}_6$ and $\mathbf{L}_7$.

$$\begin{aligned}
\mathbf{L}_6 &: \quad 1 \leq 2 \leq 3 \leq 4 \leq 5 \leq 6 \\
\mathbf{L}_7 &: \quad 1 \leq 2 \leq 3 \leq 4 \leq 5 \leq 6 \leq 7 \\
\mathbf{L}_8 &: \quad 1 \leq 2 \leq 3 \leq 4 \leq 5 \leq 6 \leq 7 \leq 8
\end{aligned}$$

Figure 5: $\mathbf{L}_6 \not\sim_3 \mathbf{L}_7$, but $\mathbf{L}_7 \sim_3 \mathbf{L}_8$.

- The Duplicator wins the 3-move Ehrenfeucht-Fraïssé game on $\mathbf{L}_7$ and $\mathbf{L}_8$.

The Spoiler can win the 3-move Ehrenfeucht-Fraïssé game on $\mathbf{L}_6$ and $\mathbf{L}_7$ by playing as follows: the first move of the Spoiler is element 4 in $\mathbf{L}_7$. In order to avoid loosing in the next move, the Duplicator has to play either element 4 in $\mathbf{L}_6$ or element 3 in $\mathbf{L}_6$. If the Duplicator plays 4 in $\mathbf{L}_6$, then the Spoiler plays element 6 in $\mathbf{L}_7$. At this point, the Duplicator must play either element 5 in $\mathbf{L}_6$ or element 6 in $\mathbf{L}_6$. In the first case, the Spoiler wins the run by playing element 5 in $\mathbf{L}_7$; in the second case, the Spoiler wins the run by playing element 7 in $\mathbf{L}_7$. An essentially symmetric argument shows that the Spoiler can win if the first move of the Duplicator is element 3 in $\mathbf{L}_6$.

In contrast, consider the 3-move Ehrenfeucht-Fraïssé game on $\mathbf{L}_7$ and $\mathbf{L}_8$, and suppose that the Spoiler plays element 4 in $\mathbf{L}_8$. In this case, the Duplicator responds by playing element 4 in $\mathbf{L}_7$. If the Spoiler plays element 6 in $\mathbf{L}_8$, then the Duplicator plays element 6 in $\mathbf{L}_7$ and after this can easily maintain a partial isomorphism no matter what the third move of the Spoiler is. Similarly, if the second move of the Spoiler is element 7 in $\mathbf{L}_8$, then the second move of the Duplicator is element 6 in $\mathbf{L}_7$. We leave it to the reader to fill in the remaining cases and verify that the Duplicator wins the 3-move Ehrenfeucht-Fraïssé game on $\mathbf{L}_7$ and $\mathbf{L}_8$. ■

We are now ready to describe the analysis of $\sim_r$, $r \geq 1$, on finite linear orders and derive the preceding Example 3.19 as a special case.

Theorem 3.20: *Let r, m, and n be positive integers. Then the following are equivalent:*

- $\mathbf{L}_m \sim_r \mathbf{L}_n$.
- *$(m = n)$ or $(m \geq 2^r - 1$ and $n \geq 2^r - 1)$*

Proof: *(Hint)* If c is an element of the linear order $\mathbf{L}_n$, then $\mathbf{L}_n^{>c}$ denotes the linear order with universe $\{d : c < d \leq n\}$ and, similarly, $\mathbf{L}_n^{<c}$ denotes the linear order with universe $\{d : 1 \leq d < c\}$. It is easy to see that for every positive integer s we have that $\mathbf{L_m} \sim_{s+1} \mathbf{L}_n$ if and only if the following two conditions hold:

1. For every $a \in \mathbf{L}_m$, there is $b \in L_n$ such that $\mathbf{L}_m^{>a} \sim_s \mathbf{L}_n^{>b}$ and $\mathbf{L}_m^{<a} \sim_s \mathbf{L}_n^{<b}$.
2. For every $b \in \mathbf{L}_n$, there is $a \in \mathbf{L}_m$ such that $\mathbf{L}_m^{>a} \sim_s \mathbf{L}_n^{>b}$ and $\mathbf{L}_m^{<a} \sim_s L_n^{<b}$.

The result can then be derived from the above fact using induction on $\min(m, n)$. ■

Corollary 3.21: *The* Even Cardinality *query is not first-order definable on the class $\mathcal{L}$ of all finite linear orders.*

Proof: Apply the method of Ehrenfeucht-Fraïssé games using the linear orders $\mathbf{L}_{2m}$ and $\mathbf{L}_{2m+1}$ with $m \geq 2^r - 1$. ∎

As indicated earlier, the class of finite linear orders provides a rather rare example of a class of structures for which a complete analysis of the $\equiv_r$-equivalence classes, $r \geq 1$, has been obtained. Over the years, however, researchers have succeeded in identifying general sufficient conditions for the Duplicator to win the r-move Ehrenfeucht-Fraïssé game. These conditions give "off-the-shelf" winning strategies for the Duplicator and thus facilitate the application of the method of Ehrenfeucht-Fraïssé games. In what follows, we will present such a useful and widely applicable sufficient condition discovered by Fagin, Stockmeyer and Vardi [FSV95], who built on earlier work by Hanf [Han65]. Additional useful sufficient conditions for the Duplicator to win the Ehrenfeucht-Fraïssé game have been found by Schwentick [Sch94], Arora and Fagin [AF97], and others (see Fagin [Fag97] for a survey). Underlying this work is Gaifman's theorem [Gai82], which intuitively asserts that first-order logic can express *local* properties only. Although we will not discuss or use Gaifman's theorem here, we will introduce the fundamental concept of *neighborhood*, which plays a key role in both Gaifman's work and the work on sufficient conditions for the Duplicator to win the Ehrenfeucht-Fraïssé game.

Definition 3.22: Let $\mathbf{A} = (A, R_1^{\mathbf{A}}, \ldots, R_m^{\mathbf{A}}, c_1^{\mathbf{A}}, \ldots, c_s^{\mathbf{A}})$ be a σ-structure, let a be an element of A, and let d be a positive integer.

- The *Gaifman graph* $\mathbf{G}_{\mathbf{A}} = (A, E_{\mathbf{A}})$ of $\mathbf{A}$ is the undirected graph having the elements of A as nodes and edge relation $E_{\mathbf{A}}$ defined as follows: there is an edge $E_{\mathbf{A}}(b, c)$ between two elements b and c of A if there is a relation $R_i^{\mathbf{A}}$, $1 \leq i \leq m$, and a tuple $(t_1, \ldots, t_s) \in R_i^{\mathbf{A}}$ such that b and c are among $t_1, \ldots, t_s$.
- The *neighborhood $N(a, d)$ of a of radius d* is the set of all nodes whose distance in the Gaifman graph $\mathbf{G}_{\mathbf{A}}$ from a or from one of the constants $c_1^{\mathbf{A}}, \ldots, c_s^{\mathbf{A}}$ is less than d. More formally, $N(a, d)$ is defined by the following induction on d:
  - $N(a, 1) = \{a, c_1^{\mathbf{A}}, \ldots, c_s^{\mathbf{A}}\}$.
  - $N(a, d+1) = N(a, d) \cup \{c \in A :$ there is a $b \in N(a, d)$ such that $E_{\mathbf{A}}(b, c)\}$. ∎

The following examples reveal that the neighborhood of an element can vary widely.

Example 3.23: Let $n \geq 1$ and $d \geq 2$ be positive integers.

- If $\mathbf{L}_n = (L_n, \leq)$ is a linear order with n elements, then $N(a, d) = L_n$, for every $a \in L_n$.
- If $\mathbf{K}_n = (K_n, E)$ is a clique with n nodes, then $N(a, d) = K_n$, for every $a \in K_n$.
- If $\overline{\mathbf{K}}_n = (\overline{K}_n, E)$ is a totally disconnected graph with n nodes, then $N(a, d) = \{a\}$, for every $a \in \overline{K}_n$.
- If $\mathbf{C}_n = (C_n, E)$ is a (directed or undirected) cycle C_n with n nodes and $d \leq n/2$, then $N(a, d)$ is an undirected path with $2d - 1$ nodes having a as its midpoint. ∎

∎

Definition 3.24: Let $\mathbf{A} = (A, R_1^{\mathbf{A}}, \ldots, R_m^{\mathbf{A}}, c_1^{\mathbf{A}}, \ldots, c_s^{\mathbf{A}})$ be a σ-structure, let a be an element of A, and let d be a positive integer.

- $(\mathbf{A}, a)$ denotes the expansion of $\mathbf{A}$ obtained by augmenting it with a as a distinguished element interpreting a new constant.
- $(\mathbf{A}, a) \restriction N(a, d)$ denotes the substructure of $(\mathbf{A}, a)$ generated by $N(a, d)$.
- The *d-type of* a is the isomorphism type of the structure $(\mathbf{A}, a) \restriction N(a, d)$. ∎

Note that the universe of $(\mathbf{A}, a) \restriction N(a, d)$ is $N(a, d)$, since $N(a, d)$ contains a, $c_1^{\mathbf{A}}, \ldots, c_s^{\mathbf{A}}$ as members. Moreover, if $\mathbf{B} = (B, R_1^{\mathbf{B}}, \ldots, R_m^{\mathbf{B}}, c_1^{\mathbf{B}}, \ldots, c_s^{\mathbf{B}})$ is a σ-structure and b is an element of B, then a and b have the same d-type precisely when there is a one-to-one and onto mapping $h : N(a, d) \to N(b, d)$ such that $h(a) = b$, $h(c_i^{\mathbf{A}}) = c_i^{\mathbf{B}}$, $1 \leq i \leq s$, and for every relation symbol R_i of arity t, $1 \leq i \leq m$, and every t-tuple $(a_1, \ldots, a_t)$ from $N(a, d)$, we have that $R_i^{\mathbf{A}}(a_1, \ldots, a_t)$ if and only if $R_i^{\mathbf{B}}(h(a_1), \ldots, h(a_t))$.

Definition 3.25: Assume that d is a positive integer, σ is a vocabulary, and $\mathbf{A}$ and $\mathbf{B}$ are two σ-structures. We say that $\mathbf{A}$ and $\mathbf{B}$ are *d-equivalent* if for every d-type τ they have the same number of elements of d-type τ. ∎

Clearly, d-equivalence is an equivalence relation on the class $\mathcal{S}$ of all σ-structures. The next result, due to Fagin, Stockmeyer and Vardi [FSV95], asserts that if d is sufficiently larger than r, then d-equivalence is actually a refinement of $\equiv_r$-equivalence.

Theorem 3.26: *For every positive integer r and for every positive integer $d \geq 3^{r-1}$, if $\mathbf{A}$ is d-equivalent to $\mathbf{B}$, then $\mathbf{A} \equiv_r \mathbf{B}$.*

Proof: *(Hint)* Assume that $\mathbf{A}$ is d-equivalent to $\mathbf{B}$, where $d \geq 3^{r-1}$. We can show that the Duplicator wins the r-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$ by maintaining a partial isomorphism between not only the elements of $\mathbf{A}$ and $\mathbf{B}$ played thus far, but also between neighborhoods of these points of sufficiently large radius. Specifically, by induction on $j \leq m$, it can be shown that the Duplicator can win the r-move Ehrenfeucht-Fraïssé-game via a winning strategy that has the following property, called the *j-matching condition*:

If $a_1, \ldots, a_j$ and $b_1, \ldots, b_j$ are the elements of $\mathbf{A}$ and $\mathbf{B}$ played in the first j moves of a run, then $\mathbf{A} \restriction \cup_{i=1}^{j} N(a_i, 3^{r-j})$ is isomorphic to $\mathbf{B} \restriction \cup_{i=1}^{j} N(b_i, 3^{r-j})$ via an isomorphism that maps a_i to b_i, for $1 \leq i \leq j$.

Note that the Duplicator can ensure that the 1-matching condition holds as follows. If the Spoiler plays an element a_1 in A (or an element b_1 in B), then, by d-equivalence, there is an element b_1 in B (or an element a_1 in A) such that a_1 and b_1 have the same d-type, which implies that $\mathbf{A} \restriction N(a_1, 3^{r-1})$ is isomorphic to $\mathbf{B} \restriction N(b_1, 3^{r-1})$ via an isomorphism that maps a_1 to b_1. The inductive step from j to $j + 1$ uses d-equivalence combined with a counting argument to the effect that the Duplicator can always find at least one element with the same d-type as the last element played by the Spoiler, but not contained in the union of neighborhoods of radius $3^{r-(j+1)}$ of the elements played so far. ∎

The preceding Theorem 3.26 gives rise to a new method for studying first-order definability.

Method 3.27: Let σ be a vocaculary, $\mathcal{C}$ a class of σ-structures, and Q a Boolean query on $\mathcal{C}$. To show that Q is *not* first-order definable on $\mathcal{C}$, it suffices to show that for every positive integer r there are structures $\mathbf{A}_r$ and $\mathbf{B}_r$ in $\mathcal{C}$ such that

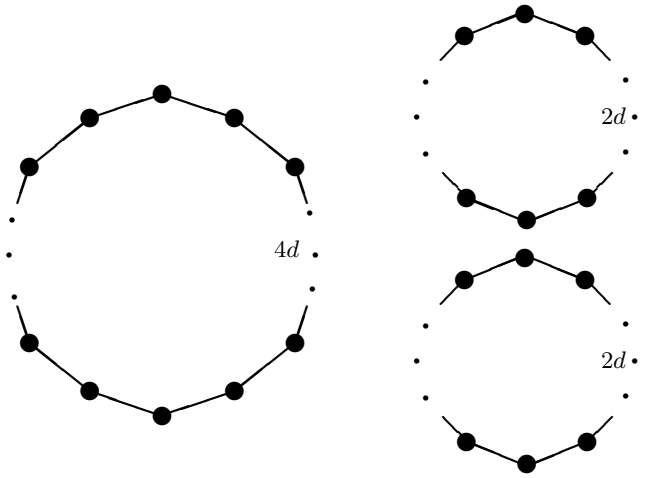
- $Q(\mathbf{A}_r) = 1$ and $Q(\mathbf{B}_r) = 0$.

- $\mathbf{A}_r$ is d-equivalent to $\mathbf{B}_r$ for some $d \geq 3^r$. ∎

Although, by Theorem 3.26, this method is sound, it is not complete. For instance, it cannot be used to analyze first-order definability on the class $\mathcal{L}$ of all finite linear orders, since, for every positive integers d, m, and n, the linear order $\mathbf{L}_m$ is d-equivalent to the linear order $\mathbf{L}_n$ if and only if $m = n$. In particular, this method cannot be used to show that EVEN CARDINALITY is not first-order definable on finite linear orders. Nonetheless, whenever applicable, Method 3.27 is usually technically simpler than Method 3.16, since it replaces the task of proving that the Duplicator wins the r-move Ehrenfeucht-Fraïssé game with the task of analyzing and counting d-types. Moreover, the analysis of d-types often provides a clue for finding candidate structures $\mathbf{A}_r$ and $\mathbf{B}_r$. In the remainder of this section, we present several applications of Method 3.27 to finite model theory.

Proposition 3.28: *The* CONNECTIVITY *query is not first-order definable on finite graphs.*

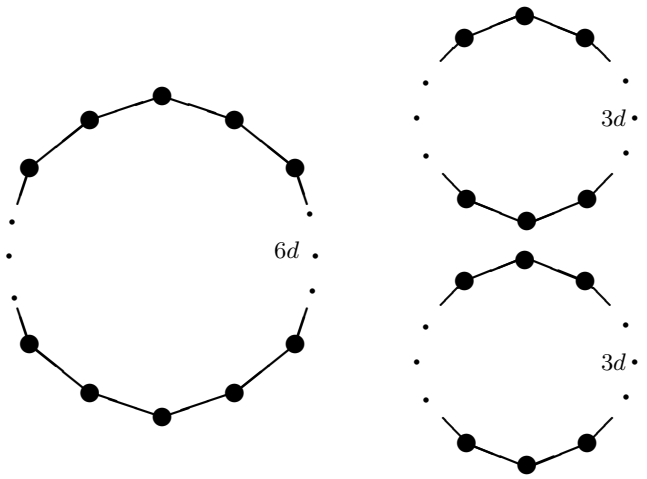
Proof: For every r and every $d \geq 3^{r-1}$, let $\mathbf{A}_r$ be a cycle with $4d$ nodes and $\mathbf{B}_r$ be the union of two disjoint cycles each with $2d$ nodes, as depicted below.



Clearly, each d-type in $\mathbf{A}_r$ or in $\mathbf{B}_d$ is a path with $2d - 1$ nodes. Moreover, $\mathbf{A}_r$ is d-equivalent to $\mathbf{B}_r$, since each structure has exactly $4d$ points of this d-type. ∎

Proposition 3.29: *The* 2-COLORABILITY *query is not first-order definable on finite graphs.*

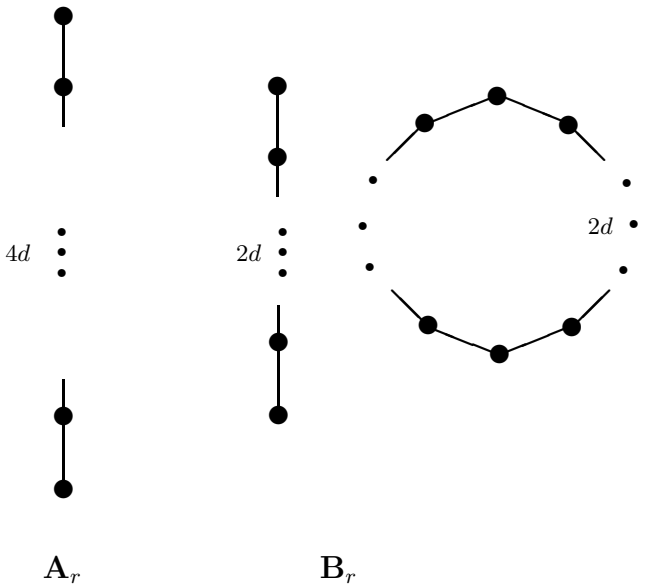
Proof: For every r, let $d = 3^{r-1}$, and let $\mathbf{A}_r$ be a cycle with $6d$ nodes and $\mathbf{B}_d$ be the union of two disjoint cycles each with $3d$ nodes, as depicted below.



$\mathbf{A}_r$ is 2-colorable, but $\mathbf{B}_r$ is not, since $\mathbf{A}_r$ is an even cycle, while $\mathbf{B}_r$ contains an odd cycle. Moreover, $\mathbf{A}_r$ is d-equivalent to $\mathbf{B}_r$. ∎

Proposition 3.30: *The* ACYCLICITY *query is not first-order definable on finite graphs.*

Proof: Let $\mathbf{A}_r$ and $\mathbf{B}_r$ be the two structures depicted.



Clearly, $\mathbf{A}_r$ is acyclic, $\mathbf{B}_r$ contains a cycle, and $\mathbf{A}_r$ is d-equivalent to $\mathbf{B}_r$. ∎

Exercise 3.31: Show that the following queries are not first-order definable on the class of all finite graphs:

- k-COLORABILITY, for each fixed $k \geq 3$
- PLANARITY

- RIGIDITY

Exercise 3.32: Show that the CONNECTIVITY query is not first-order definable on the class $\mathcal{O}$ of all finite linearly ordered graphs $\mathbf{G} = (V, E, \leq)$.

Hint: Use the analysis of $\equiv_r$-equivalence on linear orders.

4 Computational Complexity

This section is a brief interlude to computational complexity and a first encounter with the connections between computational complexity and logics on finite structures. These connections are explored in depth in the Chapter by Erich Grädel in this volume.

4.1 Complexity Classes

In his 1993 Turing Award Lecture [Har94], J. Hartmanis described computational complexity as "the quantitative study of solvability". Indeed, the main goal of computational complexity is to characterize the inherent difficulty of solvable decision problems by placing them into classes according to the time resources or space resources required to solve them in some model of computation, which usually is either the (deterministic) Turing machine or the nondeterministic Turing machine. The following major complexity classe will be of interest to us here; their precise definitions can be found in [Pap94].

Class	Resource Bound
L	Logarithmic Space
P	Polynomial Time
NP	Nondeterministic Polynomial Time
PSPACE	Polynomial Space
EXPTIME	Exponential Time
NEXPTIME	Nondeterministic Exponential Time

It is well known and easy to show that the following containments hold:

$$\mathrm{L} \subseteq \mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXPTIME} \subseteq \mathrm{NEXPTIME}.$$

It is also conjectured and widely believed that each of the above containments is a proper one, but proving this remains the central open problem in computational complexity to date. It has been established, however, that if there is an exponential gap in the amount of the resource (space or time) used in defining two complexity classes, then one is properly contained in the other. These results, which are are known as space and time *hierarchy* theorems (see [Pap94]), imply that

$$\mathrm{L} \subsetneq \mathrm{PSPACE}, \quad \mathrm{P} \subsetneq \mathrm{EXPTIME}, \quad \mathrm{NP} \subsetneq \mathrm{NEXPTIME}.$$

A possible approach to separating two complexity classes is to show that there is a structural property possessed by one class, but not by the other. Clearly, each of the deterministic classes L, P, PSPACE, and EXPTIME is closed under complements. In contrast, the class NP of all problems solvable by a nondeterministic polynomial-time bounded Turing machine is not known to be closed under complements. Thus, the question: "is NP = coNP?" constitutes another major open problem in computational complexity. The same state of affairs also holds true for the class NEXPTIME.

Each of the aforementioned complexity classes contains problems that are *complete* for the class, i.e., problems that embody the intrinsic computational difficulty of the class at hand. More precisely, let C be a complexity class and Q a decision problem. We say that Q is *C-complete* if Q is in C and Q is *C-hard*, i.e., for every $Q' \in C$, there is a "suitable" many-one reduction f of Q' to Q, so that for every input x

$$x \in Q' \Longleftrightarrow f(x) \in Q.$$

If C is the class P of all polynomial-time solvable problems, then "suitable" means that f is computable in logarithmic space. For NP and all other larger classes, "suitable" means that f is computable in polynomial time. Representative natural NP-complete problems include Boolean satisfiability SAT, 3-COLORABILITY, and HAMILTONIAN PATH (see [GJ79]). The prototypical PSPACE-complete problem is QBF, the satisfiability problem for quantified Boolean formulas [Sto77].

4.2 The Complexity of Logic

Vardi [Var82] singled out certain fundamental decision problems that arise from the analysis of the satisfaction relation between sentences of a logic L and finite structures.

Definition 4.1: ([Var82]) Let L be a logic.

- The *combined complexity* of L is the following decision problem: given a finite structure $\mathbf{A}$ and an L-sentence ψ, does $\mathbf{A} \models \psi$?
- The *data complexity* of L is the family of the following decision problems Q_ψ, one for each fixed L-sentence ψ: given a finite structure $\mathbf{A}$, does $\mathbf{A} \models \psi$?
- The *expression complexity* of L is the family of the following decision problems $Q_{\mathbf{A}}$, one for each fixed finite structure $\mathbf{A}$: given an L-sentence ψ, does $\mathbf{A} \models \psi$?

The combined complexity problem for L is also known as the *model checking* problem for L. In this problem, the input consists of both a finite structure and an L-sentence. The data complexity and the expression complexity are the restricted cases of the combined complexity problem in which an L-sentence is kept fixed or a finite structure is kept fixed, respectively. Note that the data complexity and the expression complexity are not single decision problems, but families of decision problems. The next definition provides a way to "measure" the computational complexity of these families of decision problems.

Definition 4.2: ([Var82]) Let L be a logic and C a complexity class.

- *The data complexity of L is in C* if for each L-sentence ψ, the decision problem Q_ψ is in C.
- *The data complexity of L is C-complete* if it is in C and there is at least one L-sentence ψ such that the decision problem Q_ψ is C-complete.
- *The expression complexity of L is in C* if for each finite structure $\mathbf{A}$, the decision problem $Q_{\mathbf{A}}$ is in C.
- *The expression complexity of L is C-complete* if it is in C and there is at least one finite structure $\mathbf{A}$ such that the decision problem $Q_{\mathbf{A}}$ is C-complete.

The next result pinpoints the data, expression and combined complexity of first-order logic.

Theorem 4.3: *The following hold for first-order logic* FO*.*

- *The data complexity of* FO *is in* L*.*
- *The expression complexity of* FO *is* PSPACE*-complete*
- *The combined complexity of* FO *is* PSPACE*-complete.*

Proof: (*Hint*) For simplicity, assume that ψ is a first-order sentence in prenex normal form. Given a finite structure $\mathbf{A}$, one can check whether $\mathbf{A} \models \psi$ by examining each possible instantiation of quantifiers in ψ one at a time (this requires logarithmic space), while keeping track of the number of them in binary with the aid of a counter. Since there are polynomially-many such instantiations, only logarithmically many cells are used to keep track of the counter, so the entire computation requires $O(\log(|A|)$ space, where $|A|$ is the cardinality of the universe of $\mathbf{A}$.

If the sentence ψ is part of the input, then the above computation can be carried out in space bounded by a polynomial in $|A|$, so the combined complexity of FO is in PSPACE. Finally, the expression complexity of FO is PSPACE-complete (hence, the combined complexity of FO is PSPACE-complete as well), because, for every fixed finite structure $\mathbf{A}$ with at least two distinct elements, the satisfiability problem QBF for quantified Boolean formulas is easily reducible to the expression complexity problem $Q_{\mathbf{A}}$. ∎

The preceding Theorem 4.3 shows that an exponential gap exists between the data complexity of first-order logic and the expression complexity of first-order logic, and that the expression complexity of first-order logic is as hard as the combined complexity of first-order logic. As pointed out by Vardi [Var82], this phenomenon is also encountered in several other logics studied in finite model theory.

The r-move Ehrenfeucht-Fraïssé game gives rise to the natural decision problem of determining the winner of this game. In fact, there are two versions of this problem, one in which the number of moves is fixed and one in which the number of moves is part of the input. The next two results identify the computational complexity of these problems.

Proposition 4.4: *Let r be a fixed positive integer. The following problem is in* L *and, hence, also in* P*: given two finite structures $\mathbf{A}$ and $\mathbf{B}$, does the Duplicator win the r-move Ehrenfeucht-Fraïssé game on $\mathbf{A}$ and $\mathbf{B}$?*

Proof: By Theorem 3.8, for each fixed r, there are finitely many $\equiv_r$-classes and each such class is first-order definable; moreover, the proof of Lemma 3.12 provides an explicit construction of the first-order formulas that define the $\equiv_r$-equivalence classes. The conclusion now follows from the fact that, by Theorem 4.3, the data complexity of FO is in L. ∎

Pezzoli [Pez99] established that if the number of moves is part of the input, then determining the winner of the r-move Ehrenfeucht-Fraïssé-game is a much harder task.

Theorem 4.5: *The following problem is* PSPACE*-complete: given a positive integer $r \geq 1$ and two finite structures $\mathbf{A}$ and $\mathbf{B}$, does the Duplicator win the r-move E-F game on $\mathbf{A}$ and $\mathbf{B}$?*

This result is proved via a reduction from QBF that entails the construction of rather complicated combinatorial gadgets. It should be pointed out that, unlike many other decision problems

in which integers are part of the input, here the computational complexity remains the same (PSPACE-complete) irrespective of whether the number r of moves is given in unary or in binary. The reason is that if r is bigger than $\max\{|A|, |B|\}$, then r can be replaced by $\max\{|A|, |B|\}$; moreover, this quantity is given in unary, since at least $\max\{|A|, |B|\}$-many bits are needed to encode $\mathbf{A}$ and $\mathbf{B}$.

As shown in Section 3, first-order logic has severely limited expresive power on finite graphs. In particular, none of the queries DISCONNECTIVITY, k-COLORABILITY, $k \geq 2$ and HAMILTONIAN PATH is first-order definable on the class of all finite graphs. Recall also that these queries are easily expressible in existential second-order logic ESO, one of the two syntactically simplest fragments of second-order logic. This increase in expressive power, however, is accompanied by an increase in complexity.

Proposition 4.6: *The data complexity of* ESO *is* NP*-complete.*

Proof: Let Ψ be a fixed ESO-sentence of the form $(\exists S_1) \cdots (\exists S_m)\varphi(S_1, \ldots, S_m)$, where $\varphi(S_1, \ldots, S_m)$ is a first-order sentence. Given a finite structure $\mathbf{A}$, one can check that $\mathbf{A} \models \Psi$ by first "guessing" relations $S'_1, \ldots, S'_m$ on A and then verifying that $(\mathbf{A}, S'_1, \ldots, S'_m) \models \varphi(S_1, \ldots, S_m)$. This computation can be carried out in nondeterministic polynomial time, since the size of the relations guessed is polynomial in $|A|$ and the data complexity of first-order logic is in P. Consequently, the data complexity of ESO is in NP.

Since 3-COLORABILITY is definable by a monadic ESO-sentence and it is an NP-complete problem, it follows that the data complexity of monadic ESO is NP-complete; hence, also the data complexity of ESO is NP-complete. ∎

Vardi [Var82] has shown that both the expression complexity of ESO and the combined complexity of ESO are NEXPTIME-complete; this is another instance of the exponential-gap phenomenon between the data complexity and the expression (and combined) complexity of a logic.

The link between the data complexity of ESO and NP turns out to be much stronger. The exact connection is provided by the following result, which has become known as Fagin's Theorem and constitutes the prototypical result of descriptive complexity.

Theorem 4.7: *([Fag74]) The following are equivalent for a Boolean query Q on the class $\mathcal{F}$ of all finite σ-structures.*

- *Q is in* NP*.*
- *Q is* ESO*-definable on $\mathcal{F}$.*

In other words, NP $=$ ESO *on $\mathcal{F}$.*

Fagin's Theorem asserts that in a precise sense ESO *captures* NP on the class of all finite structures and, thus, provides a logic-based and machine-independent characterization of NP. Moreover, it makes it possible to reformulate the "NP $\stackrel{?}{=}$ coNP" question in terms of logic alone.

Corollary 4.8: *The following statements are equivalent:*

- NP *is closed under complements (in other words,* NP $=$ coNP*).*
- ESO *is closed under complements on the class $\mathcal{F}$ of all finite structures (in other words,* ESO$[\mathcal{F}]$ $=$ USO$[\mathcal{F}]$*).*

- 3-COLORABILITY *is* USO-*definable on the class of all finite graphs.*
- HAMILTONIAN PATH *is* USO-*definable on the class of all finite graphs.*

Proof: The result follows from Fagin's Theorem 4.7 and the NP-completeness of 3-COLORABILITY and HAMILTONIAN PATH. ∎

5 Ehrenfeucht-Fraïssé Games for Existential Second-Order Logic

In this section, we consider certain extensions of the Ehrenfeucht-Fraïssé games that are powerful enough to characterize definability in existential second-order logic.

Definition 5.1: Let $s_1, \ldots, s_k, r$ be positive integers, σ a vocabulary, and let $\mathbf{A}$, $\mathbf{B}$ be two σ-structures. The $(\langle s_1, \ldots, s_k \rangle, r)$-*Ehrenfeucht-Fraïssé-game on* $\mathbf{A}$ *and* $\mathbf{B}$ is played according to the following rules. In a run of the game:

- The Spoiler picks relations $S_1, \ldots, S_k$ of arities $s_1, \ldots, s_k$ on A.
- The Duplicator picks relations $S'_1, \ldots, S'_k$ of arities $s_1, \ldots, s_k$ on B.
- After this, the two players engage in a run of the r-move Ehrenfeucht-Fraïssé game on the expanded structures $(\mathbf{A}, S_1, \ldots, S_k)$ and $(\mathbf{B}, S'_1, \ldots, S'_k)$.
- Let $(a_1, b_1), \ldots, (a_r, b_r)$ be the elements of $A \times B$ picked by the two players in their r moves. The Duplicator wins this run if the mapping

  $$a_i \mapsto b_i,\ 1 \leq i \leq r, \text{ and } c_i^{\mathbf{A}} \mapsto c_j^{\mathbf{B}},\ 1 \leq j \leq s,$$

  is a *partial isomorphism*, that is, an isomorphism between the substructure $(\mathbf{A}, S_1, \ldots, S_k) \restriction \{a_1, \ldots, a_r\}$ of $(\mathbf{A}, S_1, \ldots, S_k)$ generated by $\{a_1, \ldots, a_r\}$ and the substructure $(\mathbf{B}, S'_1, \ldots, S'_k) \restriction \{b_1, \ldots, b_r\}$ of $(\mathbf{B}, S'_1, \ldots, S'_k)$ generated by $\{b_1, \ldots, b_r\}$.

  Otherwise, the Spoiler wins the run.
- The *Duplicator wins the* $(\langle s_1, \ldots, s_k \rangle, r)$ *game on* $\mathbf{A}$ *and* $\mathbf{B}$ if the Duplicator can win every run of the game, i.e., if (s)he has a *winning strategy* for this game.

  Otherwise, the *Spoiler wins the* $(\langle s_1, \ldots, s_k \rangle, r)$-*Ehrenfeucht-Fraïssé game on* $\mathbf{A}$ *and* $\mathbf{B}$.

Using Theorem 3.8 and the semantics of existential second-order logic, it is quite straightforward to establish the following result.

Proposition 5.2: *Let* Ψ *be an ESO-sentence of the form* $(\exists P_1) \cdots (\exists P_k)\psi$*, where each* P_i *is a relation symbol of arity* s_i *and* ψ *is a first-order sentence of quantifier rank* r*. If* $\mathbf{A} \models \Psi$ *and the Duplicator wins the* $(\langle s_1, \ldots, s_k \rangle, r)$-*Ehrenfeucht-Fraïssé game on* $\mathbf{A}$ *and* $\mathbf{B}$*, then* $\mathbf{B} \models \Psi$*.*

In turn, this result gives rise to a combinatorial method for establishing limitations in the expressive power of existential second-order logic on arbitrary classes of structures. Moroever, it is not hard to show that the method is complete as well.

Method 5.3: The Method of Ehrenfeucht-Fraïssé Games for ESO
Let σ be a vocaculary, $\mathcal{C}$ a class of σ-structures, and Q a Boolean query on $\mathcal{C}$.

Soundness: To show that Q is *not* ESO-definable on $\mathcal{C}$, it suffices to show that for every sequence of positive integers $s_1, \ldots, s_k, r$, there are structures $\mathbf{A}$ and $\mathbf{B}$ in $\mathcal{C}$ such that

- $Q(\mathbf{A}) = 1$ and $Q(\mathbf{B}) = 0$.
- The Duplicator wins the $(\langle s_1, \ldots, s_k \rangle, r)$-*Ehrenfeucht-Fraïssé-game on* $\mathbf{A}$ *and* $\mathbf{B}$. ∎

Completeness: This method is also *complete*, i.e., if Q is not ESO-definable on $\mathcal{C}$, then for every sequence $s_1, \ldots, s_k, r$ of positive integers such structures $\mathbf{A}$ and $\mathbf{B}$ exist.

Corollary 4.8 and Method 5.3 imply that the NP $\stackrel{?}{=}$ coNP question is equivalent to a problem about combinatorial games.

Corollary 5.4: *The following statements are equivalent.*

- NP $\neq$ coNP.
- *For every* $s_1, \ldots, s_k, r$, *there are finite graphs* $\mathbf{G}$ *and* $\mathbf{H}$ *such that*
  - $\mathbf{G}$ *is not* 3-Colorable *and* $\mathbf{H}$ *is* 3-Colorable.
  - *The Duplicator wins the* $(\langle s_1, \ldots, s_k \rangle, r)$-*Ehrenfeucht-Fraïssé game game on* $\mathbf{G}$ *and* $\mathbf{H}$.

Although $(\langle s_1, \ldots, s_k \rangle, r)$-Ehrenfeucht-Fraïssé games yield a sound and complete method for studying ESO-definability (thus, potentially leading to the separation of NP from coNP), so far this approach has had rather limited successes. The reason is that formidable combinatorial difficulties arise in implementing this method when one of the integers s_i is bigger than 1, that is, when dealing with ESO-formulas in which at least one of the existentially quantified second-order variables has arity bigger than 1. Nonetheless, this method has made it possible to obtain lower bounds for definability in monadic ESO, which is the fragment of existential second-order logic that can be analyzed using $(\langle 1, 1, \ldots, 1 \rangle, r)$-Ehrenfeucht-Fraïssé games.

In certain cases, the study of ESO-definability can be made easier using a variant of the $(\langle s_1, \ldots, s_k \rangle, r)$-Ehrenfeucht-Fraïssé games that has become known as the *Ajtai-Fagin games*. In what follows in this section, we present the intuition behind the Ajtai-Fagin games and highlight some of their applications to the study of definability in monadic ESO. The first observation is that, when the method of Ehrenfeucht-Fraïssé games or one of their variants is used to show that a particular query is not definable in a certain logic, then one can expand the scope of the game and view the selection of the structures $\mathbf{A}$ and $\mathbf{B}$ as being part of the Duplicator's moves. Now, one of the main difficulties with the $(\langle s_1, \ldots, s_k \rangle, r)$-Ehrenfeucht-Fraïssé games is that in effect the Duplicator has to select the structure $\mathbf{B}$ *before* the Spoiler has picked relations $S_1, \ldots, S_k$ on $\mathbf{A}$. To bypass the Duplicator's task easier, Ajtai and Fagin [AF90] introduced a variant of the $(\langle s_1, \ldots, s_k \rangle, r)$-Ehrenfeucht-Fraïssé games in which the Duplicator selects the structure $\mathbf{B}$ *after* the Spoiler has picked relations $S_1, \ldots, S_k$ on $\mathbf{A}$. The next definition introduces the Ajtai-Fagin games for monadic ESO; it can be easily extended to games for the full ESO with notational modifications only.

Definition 5.5: Let $\mathcal{C}$ be a class of σ-structures, Q a Boolean query on $\mathcal{C}$, and k, r two positive integers. The (k, r)-*Ajtai-Fagin game for* Q *on* $\mathcal{C}$ is played according to the following rules. In a run of the game:

- The Duplicator picks a structure $\mathbf{A} \in \mathcal{C}$ such that $Q(\mathbf{A}) = 1$
- The Spoiler picks k-unary relations $S_1, \ldots, S_k$ on $\mathbf{A}$ (i.e, k subsets of A)

- The Duplicator picks a structure $\mathbf{B} \in \mathcal{C}$ such that $Q(\mathbf{B}) = 0$ and then picks k unary relations $S'_1, \ldots, S'_k$ on $\mathbf{B}$ (i.e., k subsets of B).
- After this, the two players engage in a run of the r-move Ehrenfeucht-Fraïssé game on the expanded structures $(\mathbf{A}, S_1, \ldots, S_k)$ and $(\mathbf{B}, S'_1, \ldots, S'_k)$.

The winning conditions are as in Definition 5.1.

Note that another difference between the Ajtai-Fagin games and the Ehrenfeucht-Fraïssé games considered earlier is that each Ajtai-Fagin game is defined with respect to a particular Boolean query, i.e., the query itself is one of the parameters of the game. The Ajtai-Fagin games give rise to the following method for investigating definability in monadic ESO.

Method 5.6: The Method of Ajtai-Fagin Games for monadic ESO
Let σ be a vocaculary, $\mathcal{C}$ a class of σ-structures, and Q a Boolean query on $\mathcal{C}$.

Soundness: To show that Q is *not* monadic ESO-definable on $\mathcal{C}$, it suffices to show that for every k and every r, the Duplicator wins the (k, r)-Ajtai-Fagin game for Q on $\mathcal{C}$.

Completeness: This method is also *complete*, i.e., if Q is *not* monadic ESO-definable on $\mathcal{C}$, then for every k and every r, the Duplicator wins the (k, r)-Ajtai-Fagin game for Q on $\mathcal{C}$. ∎

Fagin [Fag75] showed that the CONNECTIVITY query is not monadic ESO-definable on the class of all finite graphs using Ehrenfeucht-Fraïssé games for monadic ESO. Later on, Fagin, Stockmeyer and Vardi [FSV95] obtained a much simpler proof of this result using Ajtai-Fagin games for monadic ESO and Theorem 3.26 about d-equivalence.

Theorem 5.7: *The* CONNECTIVITY *query is not monadic ESO-definable on the class of all finite graphs.*

Proof: (*Sketch*) We will show that, for every positive integer k and every positive integer r, the Duplicator wins the (k, r)-Ajtai-Fagin game for CONNECTIVITY on the class of all finite graphs.

Suppose that k is a positive integer, $\mathbf{A} = (A, E)$ is an undirected cycle, and $S_1, \ldots, S_k$ are unary relations on A. For every node $b \in A$, we define the *color* of b to be the Boolean vector $c(b) = (c_1, \ldots, c_s)$ such that if $b \in S_i$, then $c_i = 1$; otherwise, $c_i = 0$. Note that the number of colors depends only on k. Moreover, it is easy to see that, for every $d \geq 1$ and every $a \in A$, the neighborhood $N(a, d)$ of a in $(\mathbf{A}, S_1, \ldots, S_k)$ consists of $2d-1$ points whose distance from a in $\mathbf{A}$ is at most d, and is completely determined by the colors of these points (this, of course, hinges on the fact that each S_i is a unary relation on A). Consequently, the number of different d-types depends only on k and d (and not on the cardinality $|A|$ of A).

Using these facts, we can show that the Duplicator wins the (k, r)-Ajtai-Fagin game for CONNECTIVITY on the class of all finite graphs by playing according to the following strategy.

- The Duplicator picks a large enough cycle $\mathbf{A}$, so that, for every unary relations $S_1, \ldots, S_k$ on A, there are at least $4d$ points with the same d-type in $(\mathbf{A}, S_1, \ldots, S_k)$, where $d = 3^{r-1}$.
- After the Spoiler picks unary relations $S_1, \ldots, S_k$ on A, there are two nodes a_p and a_q in A having the same d-type and are such that $N(a_p, 2d) \cap N(a_q, 2d) = \emptyset$.
- The Duplicator constructs $\mathbf{B} = \mathbf{B}_0 \oplus \mathbf{B}_1$ consisting of two disjoint cycles $\mathbf{B}_0$ and $\mathbf{B}_1$ constructed as follows:

- The Duplicator disconnects $\mathbf{A}$ by "pinching" it at a_p, a_q.
- The Duplicator creates $\mathbf{B}_0$ by joining a_p and a_{q+1} in the first component.
- The Duplicator creates $\mathbf{B}_1$ by joining a_{p+1} and a_q in the second component.

- Finally, the Duplicator picks the same unary relations $S_1, \ldots, S_k$ in $\mathbf{B}$ as the ones picked by the Spoiler on $\mathbf{A}$.

Note the structures $(\mathbf{A}, S_1, \ldots, S_k)$ and $(\mathbf{B}, S_1, \ldots, S_k)$ are d-equivalent, as each node in A has the same d-type as its "clone" in B. Consequently, the Duplicator wins the r-move Ehrenfeucht-Fraïssé game on these structures. ∎

Since the DISCONNECTIVITY query is monadic ESO-definable on the class of all finite graphs, we obtain the following separation between monadic ESO and monadic USO on finite graphs.

Corollary 5.8: *Monadic* ESO *is not closed under complements on the class of all finite graphs.*

It should be pointed out that Theorem 5.7 and Corollary 5.8 do not have any implications for the NP $\stackrel{?}{=}$ coNP problem, because CONNECTIVITY is a polynomial-time computable query and monadic ESO cannot express all NP queries. Any breakthroughs towards the separation of NP from coNP using combinatorial games will entail proving limitations in the expressive power of existential second-order formulas in which the existentially quantified second-order variables have arity bigger than one. So far, however, the successes of combinatorial games have been essentially limited to monadic existential second-order logic. In particular, the following test problem is open.

Problem 5.9: Show that there is a Boolean query Q on finite graphs such that

- Q is in NP (hence, Q is ESO-definable).
- Q is not binary ESO-definable, i.e., Q is not definable by any ESO-sentence $(\exists P_1) \cdots (\exists P_k)\psi$, where each P_i is a binary relation symbol.

Nonetheless, combinatorial games have been successfully used to establish limitations in the expressive power of monadic ESO over the class of finite graphs with "built-in" predicates, such as a *successor* relation (de Rougemont [dR87]) or a *total order* (Schwentick [Sch94]). Such results are viewed as the first stepping stone towards analyzing definability in binary existential second-order logic.

Theorem 5.10: *The* CONNECTIVITY *query is not monadic* ESO*-definable on the class of finite structures with successor, i.e., on finite structures of the form* $\mathbf{G} = (V, E, \mathrm{Suc})$*, where* E *is a binary relation on* V *and* Suc *is the graph of a successor function on* V*.*

Theorem 5.11: *The* CONNECTIVITY *query is not monadic* ESO*-definable on the class of finite ordered graphs, i.e., finite structures of the form* $\mathbf{G} = (V, E, \leq)$*, where* E *is a binary relation on* V *and* $\leq$ *is a linear order on* V*.*

6 Logics with Fixed-Point Operators

In Section 3, we used Ehrenfeucht-Fraïssé-games to establish that first-order logic has severely limited expressive power on the class $\mathcal{G}$ of all finite graphs; in particular, first-order logic fails to express such basic polynomial-time computable queries as Transitive Closure, Acyclicity, 2-Colorability, Eulerian, and Planarity. Several different mechanisms can be used to augment the syntax of first-order logic, so that the resulting logic has strictly higher expressive power of finite structures. We already saw that second-order quantification is such a mechanism. In fact, Fagin's Theorem 4.7 calibrates the exact gain in expressive power that is achieved when only existential second-order quantification in prefix form is allowed; moreover, it implies that, unless P = NP, even the syntactically simplest fragments of second-order logic can express queries that are not polynomial-time computable.

As mentioned earlier, the limited expressive power of first-order logic on finite graphs can be interpreted as an inability to express *recursion*. This realization suggests that higher expressive power can also be achieved by augmenting the syntax of first-order logic with mechanisms that embody recursion. Perhaps the most natural such mechanism is to use fixed-points of operators that describe recursive specifications; this approach has been used fruifully in many different areas of computer science, including computability theory, logic programming and denotational semantics of programming languages. As a motivating example, let us consider the *factorial* function $f(n)$, which is usually defined inductively as

$$\left| \begin{array}{lcl} f(o) & = & 1 \\ f(n+1) & = & (n+1)f(n) \end{array} \right.$$

Alternatively, the factorial function can be defined as a fixed-point of the recursive specification

$$f \quad = \quad \lambda n.(n = 0 \rightarrow 1 \,\square\, (n+1)f(n)).$$

Observe that the building blocks of the above recursive specification are operations on functions, such as definition by cases and multiplication. Here, we are interested in developing a formalism for specifying queries recursively. The key idea is to describe recursive specifications using formulas of first-order logic and then augment the syntax of first-order logic with fixed-points of such specifications. Before making this idea precise, we need to develop the basics of fixed-point theory.

6.1 Operators and Fixed-Points

Let A be a set and k a positive integer. A k-ary *operator* on A is a mapping $\Phi : \mathcal{P}(A^k) \rightarrow \mathcal{P}(A^k)$, where $\mathcal{P}(A^k)$ is the powerset of A^k (that is, the set of all k-ary relations on the universe A of $\mathbf{A}$).

A k-ary relation P is a *fixed-point* of the operator Φ if $P = \Phi(P)$. Thus, every fixed-point of Φ satisfies the recursive specification

$$(x_1, \ldots, x_k) \in P \Longleftrightarrow (x_1, \ldots, x_k) \in \Phi(P).$$

An operator may have no fixed points whatsoever or it may have more than one fixed-point. For instance, the unary operator $\Phi(P) = \overline{P}$, where $\overline{P}$ is the complement of P, has no fixed-points. In contrast, let Φ be the binary operator such that if $\mathbf{G} = (V, E)$ is a graph and P is a binary relation on V, then

$$\Phi(P) = \{(a, b) : \mathbf{G} \models E(a, b) \vee P(a, b) \vee (\exists z)(E(a, z) \wedge P(z, b))\}.$$

This operator may have several fixed-points, since every transitive relation P containing the edge relation is a fixed-point of it.

A k-ary relation P^* is the *least fixed-point* of Φ if P^* is a fixed-point of Φ and for every fixed-point P of Φ, we have that $P^* \subseteq P$. We write $\mathbf{lfp}(\Phi)$ to denote the least fixed-point of Φ (if it exists). For instance, if Φ is the above binary operator on graphs $\mathbf{G} = (V, E)$, then $\mathbf{lfp}(\Phi)$ is the transitive closure of the relation E. The property of having a least fixed-point is shared by every operator which is *monotone*; furthermore, the least fixed-point of a monotone operator can be obtained by iterating the operator. We now spell out these concepts and facts in precise terms.

Definition 6.1: Let $\Phi : \mathcal{P}(A^k) \to \mathcal{P}(A^k)$ be a k-ary operator on a set A.

- The *finite stages* Φ^n, $n \geq 1$, of Φ are defined by the induction:

$$\left| \begin{array}{lcl} \Phi^1 & = & \Phi(\emptyset) \\ \Phi^{n+1} & = & \Phi(\Phi^n). \end{array} \right.$$

  In general, for every ordinal α, the *stage* Φ^α of Φ is defined by the transfinite induction

$$\Phi^\alpha \quad = \quad \Phi(\textstyle\bigcup_{\beta<\alpha} \Phi^\beta).$$

  We put $\Phi^\infty = \bigcup_\alpha \Phi^\alpha$ for the union of all stages of Φ.

- The operator Φ is *monotone* if for every two k-ary relations P_1, P_2 on A such that $P_1 \subseteq P_2$, we have that $\Phi(P_1) \subseteq \Phi(P_2)$.

The next result, which is known as the Knaster-Tarski Theorem, describes the fundamental properties of monotone operators.

Theorem 6.2: [Kna28, Tar55] *Let Φ be a monotone k-ary operator on a set A.*

- *Φ has a least fixed-point $\mathbf{lfp}(\Phi)$.*

- *There is an ordinal $\gamma < |A^k|^+$, where $|A^k|^+$ is the smallest cardinal greater than the cardinal $|A^k|$ of A^k, such that*

$$\mathbf{lfp}(\Phi) = \Phi^\infty = \Phi^\gamma = \Phi^\delta, \text{ for every } \delta > \gamma.$$

  In particular, if A is a finite set, then there is an integer $s \leq |A|^k$ such that

$$\mathbf{lfp}(\Phi) = \Phi^\infty = \Phi^s = \Phi^\delta, \text{ for every } \delta > s.$$

- *The least fixed-point of Φ is equal to the intersection of all fixed-points of Φ.*

Proof: Since Φ is monotone, it is easy to show by transfinite induction that the sequence of stages is increasing, that is, if $\alpha < \beta$, then $\Phi^\alpha \subseteq \Phi^\beta$. Since each Φ^α is a k-ary relation on A, it has at most $|A^k|$ elements. It follows that there must exist an ordinal $\gamma < |A^k|^+$ such that $\Phi^\gamma = \Phi^{\gamma+1}$. Consequently, Φ^γ is a fixed-point of Φ and also $\Phi^\infty = \Phi^\gamma = \Phi^\delta$, for every $\delta > \gamma$. Moreover, using the monotonicity of Φ again, it is easy to show by transfinite induction that if P is a fixed-point of Φ, then $\Phi^\alpha \subseteq P$, for every α. Consequently, Φ^δ is the least fixed-point $\mathbf{lfp}(\Phi)$ of Φ, and also the intersection of all of its fixed-points. ∎

Definition 6.3: Let Φ be a monotone k-ary operator on a set A. The *closure ordinal* of Φ, denoted by $\mathrm{cl}(\Phi)$, is the smallest ordinal ordinal γ such that $\Phi^\gamma = \bigcup_{\beta<\gamma} \Phi^\beta$.

Note that if A is a finite set, then $\mathrm{cl}(\Phi)$ is a positive integer.

Let Φ be a k-ary operator on a set A. A k-ary relation P^* is the *greatest fixed-point* of Φ if P^* is a fixed-point of Φ and for every fixed-point P of Φ, we have that $P \subseteq P^*$. We write $\mathbf{gfp}(\Phi)$ to denote the greatest fixed-point of Φ (if it exists). Every monotone operator has a greatest fixed-point that can be obtained via an iteration that is *dual* to the iteration used to obtain the least fixed-point of the operator. Specifically, the *dual stages* Φ_α of Φ, where α is an ordinal, are defined by the transfinite induction:

$$\left| \begin{array}{lcl} \Phi_1 & = & \Phi(A^k) \\ \Phi_\alpha & = & \Phi(\bigcap_{\beta<\alpha} \Phi_\beta). \end{array} \right.$$

We also put $\Phi_\infty = \bigcap_\alpha \Phi_\alpha$ for the intersection of all dual stages of Φ. If Φ is monotone, then its greatest fixed-point $\mathbf{gfp}(\Phi)$ is equal to Φ_∞ and also equal to the union of all fixed-points of Φ. Moreover, there is an ordinal $\gamma < |A^k|^+$ such that

$$\mathbf{gfp}(\Phi) = \Phi_\infty = \Phi_\gamma = \Phi_\delta, \text{ for every } \delta > \gamma.$$

The duality relationship between the least fixed-point and the greatest fixed-point of a monotone k-ary operator Φ can also be seen by considering the the *dual* operator $\breve{\Phi}$ of Φ, where $\breve{\Phi}(P) = \overline{\Phi(\overline{P})}$ and $\overline{P} = A^k - P$ is the complement of P. If Φ is a monotone operator, then so is its dual $\breve{\Phi}$. Moreover, using transfinite induction, it is easy to show that $\breve{\Phi}^\alpha = \overline{\Phi_\alpha}$, for every ordinal α. Consequently, $\mathbf{gfp}(\Phi) = \overline{\mathbf{lfp}(\breve{\Phi})}$. Similarly, it is easy to show that $\mathbf{lfp}(\Phi) = \overline{\mathbf{gfp}(\breve{\Phi})}$.

As an example of an operator with interesting greatest fixed-points, let Φ be the binary operator such that if $\mathbf{G} = (V, E)$ is a graph and P is a binary relation on V, then

$$\begin{array}{rcl} \Phi(P) & = & \{(a,b) : \mathbf{G} \models (\forall a')(E(a,a') \rightarrow (\exists b')(E(b,b') \wedge P(a',b')) \wedge \\ & & \qquad (\forall b')(E(b,b') \rightarrow (\exists a')(E(a,a') \wedge P(a',b')))\}. \end{array}$$

The greatest fixed-point $\mathbf{gfp}(\Phi)$ of Φ is the greatest *bisimulation* relation on the $\mathbf{G} = (V, E)$; the concept of bisimulation plays an important role in modal logic [vB84] and also in the semantics of concurrent processes [Mil90]. The same example can also be used to illustrate the concept of the dual operator $\breve{\Phi}$ of Φ, which in this case is defined by

$$\begin{array}{rcl} \breve{\Phi}(P) & = & \{(a,b) : \mathbf{G} \models (\exists a')(E(a,a') \wedge (\forall b')(E(b,b') \rightarrow P(a',b')) \wedge \\ & & \qquad (\exists b')(E(b,b') \wedge (\forall a')(E(a,a') \rightarrow P(a',b')))\}. \end{array}$$

In the sequel, we will focus on operators that are definable using formulas of some logical formalism. Let σ be a vocabulary, S a k-ary relation symbol not in σ, and $\varphi(x_1, \ldots, x_k, S)$ a formula of some logic over the vocabulary $\sigma \cup \{S\}$ with free variables among $x_1, \ldots, x_k$. On every σ-structure $\mathbf{A}$, the formula $\varphi(x_1, \ldots, x_k, S)$ gives rise to the k-ary operator $\Phi : \mathcal{P}(A^k) \rightarrow \mathcal{P}(A^k)$ such that if P is a k-ary relation on A, then

$$\Phi(P) \; = \; \{(a_1, \ldots, a_k) : \mathbf{A} \models \varphi(a_1, \ldots, a_k, P)\}.$$

For instance, both the operator whose least fixed-point is the transitive closure of the edge relation E of a graph $\mathbf{G} = (V, E)$ and the operator whose greatest fixed-point is the greatest bisimulation relation on $\mathbf{G} = (V, E)$ are definable using first-order formulas. In what follows, when we will use the terms "the least fixed-point of a formula" and the "greatest fixed-point of a formula" for the least fixed-point and the greatest fixed-point of the operator associated with the formula. Similarly, we will use the term "the closure ordinal of a formula" for the closure ordinal of the operator associated with the formula, and we will denote it by $\mathrm{cl}(\varphi)$.

Operators also arise from formulas with *parameters*. Specifically, assume that σ is a vocabulary, $S_1, S_2, \ldots, S_m$ are relation symbols not in σ, and $\varphi(x_1, \ldots, x_k, y_1, \ldots, y_n, S_1, \ldots, S_m)$ is a formula of some logic over the vocabulary $\sigma \cup \{S_1, \ldots, S_m\}$ with free variables among $x_1, \ldots, x_k, y_1, \ldots, y_n$. Assume also that the arity of the relation symbol S_i is equal to k. For every σ-structure $\mathbf{A}$, every sequence $b_1, \ldots, b_n$ of elements from the universe A of $\mathbf{A}$, and every sequence $T_1, \ldots, T_{i-1}, T_{i+1}, \ldots, T_m$ of relations on A whose arities match those of $S_1, \ldots, S_{i-1}, S_{i+1}, \ldots, S_m$, the formula φ gives rise to the k-ary operator $\Phi : \mathcal{P}(A^k) \to \mathcal{P}(A^k)$ such

$$\Phi(P) \;=\; \{(a_1, \ldots, a_k) : \mathbf{A} \models \varphi(a_1, \ldots, a_k, b_1, \ldots, b_n, T_1, \ldots, T_{i-1}, P, T_{i+1}, \ldots, T_m)\}.$$

Note that operators with parameters can also be thought of as operators (without parameters) on structures expanded with the given parameters.

As an example, let $\varphi(x_1, y_1, S)$ be the first-order formula $E(y_1, x_1) \vee (\exists z)(S(z) \wedge E(z, x_1))$ in which y_1 is a parameter. If $\mathbf{G} = (V, E)$ is a graph and a is a node in V, then this formula gives rise to the unary operator Φ such that

$$\Phi(P) = \{b : \mathbf{G} \models E(a, b) \vee (\exists z)(P(z) \wedge E(z, b))\}.$$

Then the least-fixed point $\mathbf{lfp}(\Phi)$ consists of all nodes b in V that are reachable from a. Similarly, let $\varphi(x_1, y_1, S_1, S_2)$ be the first-order formula $E(y_1, x_1) \vee (\exists z)(S_1(z) \wedge S_2(z) \wedge E(z, x_1))$ in which y_1 and S_2 are parameters. If $\mathbf{G} = (V, E)$ is a graph, a is a node in V, and T is a subset of V, then this formula gives rise to the unary operator Φ such that

$$\Phi(P) = \{b : \mathbf{G} \models E(a, b) \vee (\exists z)(P(z) \wedge T(z) \wedge E(z, b))\}.$$

Then the least-fixed point $\mathbf{lfp}(\Phi)$ consists of all nodes b in V that are reachable from a via a path in which every intermediate node is in T.

It is easy to see that if $\varphi(x_1, \ldots, x_k, S)$ is an arbitrary first-order formula over the vocabulary $\sigma \cup \{S\}$, then, for every $n \geq 1$, there is a first-order formula $\varphi^n(x_1, \ldots, x_k)$ over the vocabulary σ such that it defines the n-th stage Φ^n of the operator Φ associated with $\varphi(x_1, \ldots, x_k, S)$ on every σ-structure $\mathbf{A}$. Consequently, if $\varphi(x_1, \ldots, x_k, S)$ is a first-order formula such that the associated operator Φ is monotone on every finite σ-structures, then for every finite σ-structure $\mathbf{A}$, there is an integer s such that the least fixed-point $\mathbf{lfp}(\Phi)$ of Φ is definable by $\varphi^s(x_1, \ldots, x_k)$ on $\mathbf{A}$. In general, however, this integer depends on $\mathbf{A}$, and there may be no integer s such that $\varphi^s(x_1, \ldots, x_k)$ defines the least fixed-point $\mathbf{lfp}(\Phi)$ of Φ on every finite σ-structure, because $\mathbf{lfp}(\Phi)$ may not be first-order definable. For instance, if $\varphi(x, y, S)$ is the formula $E(x, y) \vee (\exists z)(E(x, z) \wedge S(z, y))$, then the least fixed-point $\mathbf{lfp}(\Phi)$ is the transitive closure of E; moreover, for every $n \geq 1$, $\varphi^n(x, y)$ is a first-order formula asserting that there is a path of length at most n from x to y. This formula defines the transitive closure of E on every finite graph of diameter at most n, but, as we have seen earlier, there is no first-order formula that defines the transitive closure of E on every finite graph.

6.2 Least Fixed-Point Logic

We now examine how to augment the syntax of first-order logic with least fixed-points and greatest fixed-points of operators definable by logical formulas. Since we want our operators to be monotone, it is natural to focus on formulas that give rise to monotone operators.

Let σ be a vocabulary, $\mathcal{C}$ a class of σ-structures, and $\varphi(x_1, \ldots, x_k, S)$ a formula of some logic over the vocabulary $\sigma \cup \{S\}$, where S is a k-ary relation symbol and the free variables

of φ are among $x_1, \ldots, x_k$. We say that $\varphi(x_1, \ldots, x_k, S)$ is *monotone on* $\mathcal{C}$ if for every structure $\mathbf{A} \in \mathcal{C}$, the operator Φ associated with $\varphi(x_1, \ldots, x_k, S)$ is monotone. More generally, assume that $\varphi(x_1, \ldots, x_k, y_1, \ldots, y_n, S, S_1, \ldots, S_m)$ be a formula of some logic over the vocabulary $\sigma \cup \{S, S_1, \ldots, S_m\}$, where S is a k-ary relation symbol and the free variables of φ are among $x_1, \ldots, x_k, y_1, \ldots, y_n$. We say that $\varphi(x_1, \ldots, x_k, y_1, \ldots, y_n, S, S_1, \ldots, S_m)$ is *monotone on* $\mathcal{C}$ if for every structure $\mathbf{A} \in \mathcal{C}$, every sequence $b_1, \ldots, b_n$ of elements from the universe A of $\mathbf{A}$, and every sequence $T_1, \ldots, T_m$ of relations on A whose arities match those of $S_1, \ldots, S_m$, the operator Φ associated with the formula $\varphi(x_1, \ldots, x_k, y_1, \ldots, y_n, S, S_1, \ldots, S_m)$ and the parameters $b_1 \ldots, b_n, T_1, \ldots, T_m$ is monotone.

So, it is tempting to consider augmenting the syntax of first-order logic with the least fixed-points and the greatest fixed-points of first-order formulas that are monotone on the class $\mathcal{F}$ of all finite σ-structures. Serious difficulties arise, however, in doing so. Specifically, it is known there is no algorithm for testing whether a given first-order formula is monotone on $\mathcal{F}$ [AG87]. Consequently, if the syntax of first-order logic is augmented with the least fixed-points of first-order formulas that are monotone on $\mathcal{F}$, then the resulting logic does not have an effective syntax. One way to bypass this obstacle is to restrict attention on *positive* formulas, since positivity is a syntactic property of formulas that implies monotonicity and is easily checkable. More precisely, let $\varphi(S)$ be a first-order formula over a vocabulary containing a k-ary relation symbol S. We say that $\varphi(S)$ is *positive in* S if every occurrence of S in $\varphi(S)$ is within an even number of negations. Equivalently, a first-order formula $\varphi(S)$ is positive in S if and only if after all occurrences of the negation symbol in $\varphi(S)$ are "pushed inside", no occurrence of S is negated in the resulting formula. It is easy to verify that if $\varphi(x_1, \ldots, x_k, S)$ is positive in S and the free variables of $\varphi(x_1, \ldots, x_k, S)$ are among $x_1, \ldots, x_k$, then it is monotone on the class $\mathcal{S}$ of all σ-structures (finite and infinite). Moreover, there is a linear-time algorithm for testing whether a given first-order formula is positive. At this point, it is also worth recalling a classical result in mathematical logic to the effect that if a first-order formula is monotone on the class $\mathcal{S}$ of all σ-structures (finite and infinite), then it is logically equivalent to a positive first-order formula. Thus, positivity is a syntactic property of first-order formulas that, up to logical equivalence, exhausts the semantic property of monotonicity of first-order formulas on $\mathcal{S}$.

In view of the above, we will augment the syntax of first-order logic with the least fixed-points and the greatest fixed-point of operators definable by positive first-order formulas. However, in order to obtain a logic whose syntax is closed under the formation rules used, we will close the syntax under applications of the operations of first-order logic (that is, Boolean connectives and first-order quantification) and also under applications of least fixed-points and greatest fixed-points of positive formulas, where, as with first-order logic, a formula $\varphi(S)$ of the extended formalism is positive in a relation symbol S if every occurrence of S in $\varphi(S)$ is within an even number of negations. The resulting logic is *least fixed-point logic* LFP, whose precise syntax and semantics are given in the next definition.

Definition 6.4: Let σ be a vocabulary and let $S_1, \ldots, S_n, \ldots$ be a sequence of relation symbols such that for every $m \geq 1$, this sequence contains infinitely many relation symbols of arity m.

LFP Syntax The collection of LFP-formulas over σ is defined inductively as follows:

- Every atomic formula θ over $\sigma \cup \{S_1, \ldots, S_n, \ldots\}$ is a LFP-formula. The set free(θ) is the union of the set of all first-order variables occurring in θ and the set of all relation symbols S_i occurring in θ.

- If φ and ψ are LFP-formulas, then so are $\neg\varphi$, $\varphi\wedge\psi$, $\varphi\vee\psi$. Moreover, $\text{free}(\neg\varphi) = \text{free}(\varphi)$, $\text{free}(\varphi \wedge \psi) = \text{free}(\varphi \vee \psi) = \text{free}(\varphi) \cup \text{free}(\psi)$.
- If φ is a LFP-formula and x is a first-order variable, then $\exists x\varphi$ and $\forall x\varphi$ are LFP-formulas. Moreover, $\text{free}(\exists x\varphi) = \text{free}(\forall x\varphi) = \text{free}(\varphi) \setminus \{x\}$.
- Assume that φ is a LFP-formula, S_i is a k-ary relation symbol in $\text{free}(\varphi)$ which is *positive* in φ (that is, every occurrence of S_i in φ is within an even number of negation symbols), $\mathbf{x} = (x_1, \ldots, x_k)$ is a k-tuple of first-order variables each of which is in $\text{free}(\varphi)$, and $\mathbf{u} = (u_1, \ldots, u_k)$ is a k-tuple of first-order variables not occuring in φ. Then the expressions $[\mathbf{lfp}\ S_i\mathbf{x}.\varphi](\mathbf{u})$ and $[\mathbf{gfp}\ S_i\mathbf{x}.\varphi](\mathbf{u})$ are LFP-formulas. Moreover, $\text{free}([\mathbf{lfp}\ S_i\mathbf{x}.\varphi](\mathbf{u})) = \text{free}([\mathbf{gfp}\ S_i\mathbf{x}.\varphi](\mathbf{u})) = (\text{free}(\varphi) \setminus \{x_1, \ldots, x_k, S_i\}) \cup \{u_1, \ldots, u_k\}$.

  Notation: If φ is such that S_i is the only relation symbol from $S_1, \ldots, S_n, \ldots$ that occurs free in φ and all free first-order variables of φ are among $\mathbf{x} = (x_1, \ldots, x_k)$, then we will often write $\varphi^\infty(\mathbf{u})$ instead of $[\mathbf{lfp}\ S_i\mathbf{x}.\varphi](\mathbf{u})$.

LFP Semantics The semantics of least fixed logic is defined by a straightforward induction on the construction of LFP-formulas. For instance, the semantics of $[\mathbf{lfp}\ S_i\mathbf{x}.\varphi](\mathbf{u})$ is the least fixed-point of the operator associated with φ on a σ-structure $\mathbf{A}$ and parameters from $\mathbf{A}$ corresponding to the first-order variables and relation symbols in $\text{free}([\mathbf{lfp}\ S_i\mathbf{x}.\varphi](\mathbf{u}))$. Specifically, assume that φ is a LFP-formula such that $\text{free}(\varphi) \subseteq \{x_1, \ldots, x_k, y_1, \ldots, y_n, S_1, \ldots, S_m\}$ and S_i is a k-ary relation symbol that is positive in φ. Put $\mathbf{x} = (x_1, \ldots, x_k)$, $\mathbf{y} = (y_1, \ldots, y_n)$. Let $\mathbf{A}$ be a σ-structure, $\mathbf{a}$ a k-tuple from A, $\mathbf{b}$ a n-tuple from A, and $T_1, \ldots, T_{i-1}, T_{i+1}, \ldots, T_m$ relations on A whose arities match those of the relation symbols $S_1, \ldots, S_{i-1}, S_{i+1}, \ldots S_m$. Then $\mathbf{A}, \mathbf{a} \models [\mathbf{lfp}\ S_i\mathbf{x}.\varphi](\mathbf{u})$ if $\mathbf{a} \in \mathbf{lfp}(\Phi)$, where Φ is the k-ary operator on A such that

$$\Phi(P) = \{\mathbf{a} \in A^k : \mathbf{A} \models \varphi(\mathbf{a}, \mathbf{b}, T_1, \ldots, T_{i-1}, P, T_{i+1}, \ldots, T_m)\}.$$

Similarly, the semantics of $[\mathbf{gfp}\ S_i\mathbf{x}.\varphi](\mathbf{u})$ is the greatest fixed-point of the operator associated with φ on a σ-structure $\mathbf{A}$ and parameters from $\mathbf{A}$ corresponding to the first-order variables and and relation symbols in $\text{free}([\mathbf{gfp}\ S_i\mathbf{x}.\varphi](\mathbf{u}))$.

As an example, if $\varphi(x_1, y_1, S_1, S_2)$ is the first-order formula $E(y_1, x_1) \vee (\exists z)(S_1(z) \wedge S_2(z) \wedge E(z, x_1))$, then for every graph $\mathbf{G} = (V, E)$, every node a in V, and every subset T of V, the LFP-formula $[\mathbf{lfp}\ S_1(x_1).\varphi](u)$ defines the set of nodes u reachable from a via a path in which every intermediate node is in T.

The syntax of least fixed-point logic LFP, as presented in Definition 6.4, allows for arbitrary nesting of least fixed-points and greatest fixed-points, as well as for the interleaving of least and greatest fixed-points with the operations of first-order logic. Although the full syntax of LFP will be used in other chapters in this volume, in the remainder of this chapter we will focus on LFP_1, which is one of the syntactically simplest and most well-studied fragments of LFP. Informally, LFP_1 is the extension of first-order logic obtained by augmenting the syntax of first-order logic with the least fixed-points of positive formulas (without parameters) and then closing under conjunctions, disjunctions, existential and universal first-order quantification. The precise definition of LFP_1 follows.

Definition 6.5: Let σ be a vocabulary.

The collection of LFP_1-formulas over σ is defined inductively as follows:

- Every first-order formula over σ is a LFP_1-formula over σ.

- If k is a positive integer, S is a k-ary relation symbol not in σ, and $\varphi(x_1, \ldots, x_k, S)$ is a positive in S first-order formula over the vocabulary $\sigma \cup \{S\}$, and $u_1, \ldots, u_k$ are first-order variables, then the expression $[\mathbf{lfp}\ S\mathbf{x}.\varphi](\mathbf{u})$ is a LFP_1-formula, where $\mathbf{x} = (x_1, \ldots, x_k)$ and $\mathbf{u} = (u_1, \ldots, u_k)$. Since the $\varphi(x_1, \ldots, x_k, S)$ contains no parameters, in the sequel we will use the expression $\varphi^\infty(u_1 \ldots, u_k)$ to denote the formula $[\mathbf{lfp}\ S\mathbf{x}.\varphi](\mathbf{u})$.
- If φ and ψ are LFP_1-formulas over σ, then $\varphi \wedge \psi$ and $\varphi \vee \psi$ are LFP_1-formulas over σ.
- If ψ is a LFP_1-formula over σ and x is a first-order variable, then $\exists x\psi$ and $\forall x\psi$ are LFP1-formulas over σ.

Since every LFP_1-formula is a LFP-formula, the semantics of LFP_1 is inherited from the semantics of LFP.

The study of LFP_1 -definable relations on fixed infinite structures is the focus of Moschovakis' monograph "Elementary Induction on Abstract Structures" [Mos74], where they are called *inductive* relations. It should also be pointed out that in Immerman's book on "Descriptive Complexity" LFP is denoted by FO(LFP) (the closure of FO under least fixed-points) and LFP_1 is denoted by LFP(FO) (least fixed-points of first-order formulas).

Note that LFP_1-formulas are closed under the positive operations of first-order logic, but they are not closed under negation. Consequently, for every class $\mathcal{C}$ of σ-structures, it is an interesting problem to determine whether or not the collection of LFP_1-definable queries on $\mathcal{C}$ is closed under complements. In what follows, we will explore the expressive power of LFP_1 on the class $\mathcal{F}$ of all finite σ-structures and we will also study the complementation problem for LFP_1-definable queries on $\mathcal{F}$. We begin by presenting several examples that illustrate the expressive power of LFP_1 on finite structures.

Example 6.6: Transitive Closure and Connectivity.

Let $\varphi(x, y, S)$ be the positive in S existential first-order formula

$$E(x, y) \vee (\exists z)(E(x, z) \wedge S(z, y)).$$

As seen earlier, $\varphi^\infty(x, y)$ defines the Transitive Closure query TC on the class of all graphs $\mathbf{G} = (V, E)$. Thus, TC is an example of a query that is LFP_1-definable, but not FO-definable. Note that for every graph $\mathbf{G} = (V, E)$ (finite or infinite), we have that $\mathrm{cl}(\varphi) \leq \omega$.

Observe that the LFP_1-formula $(\forall x)(\forall y)\varphi^\infty(x, y)$ defines the Connectivity query CN on the class of all graphs; this gives another example of a query that is LFP_1-definable, but not FO-definable.

If $\psi(x, y, S)$ is the positive in S existential first-order formula

$$E(x, y) \vee (\exists z)(S(x, z) \wedge S(z, y)),$$

then $\psi^\infty(x, y)$ is a LFP_1-formula that also defines the Transitive Closure query TC on the class of all graphs. Although $\varphi(x, y, S)$ and $\psi(x, y, S)$ have the same least fixed-points, their stages behave differently. Specifically, for each $n \geq 1$, the n-th stage $\varphi^n(x, y)$ defines all pairs of nodes that are connected via a path of length at most n, while the n-th stage $\psi^n(x, y)$ defines all pairs of nodes that are connectd via a path of length at most 2^n. Thus, on a finite structure $\mathbf{A}$, we have that $\mathrm{cl}(\varphi) \leq |A|$, while $\mathrm{cl}(\psi) \leq \log(|A|)$. ∎

Example 6.7: PATH SYSTEMS

Let σ be a vocabulary consisting of a unary relation symbol and a ternary relation symbol. Thus, a σ-structure is a structure of the form $\mathbf{S} = (F, A, R)$, where A is a subset of F and R is a ternary relation on F. Such structures can be thought of as encoding *proof systems* in which F is a set of *formulas*, A is a set of *axioms*, and R is a ternary *rule of inference*, such as modus ponens or resolution (that is, $R(f, g, h)$ means that f can be derived from g and h using rule R). In this framework, a formula $f \in F$ is a *theorem* of S if either f is one of the axioms in A or it can be derived from other previously derived theorems g and h of $\mathbf{S}$ using the rule of inference R.

The following unary query, called PATH SYSTEM, arises naturally now: given a finite σ-structure $\mathbf{S} = (F, A, R)$ and a formula $f \in F$, find the set of all theorems of S. The computational complexity of this query was investigated by Cook [Coo74], who showed that it is P-complete under logarithmic space reductions. In fact, this was the first problem shown to be complete for polynomial-time computability, and its discovery gave rise to the theory of P-completeness (see [GHR95]).

Using Ehrenfeucht-Fraïssé-games, it can be proved that PATH SYSTEMS is not FO-definable. It is easy to see, however, that PATH SYSTEMS is LFP_1-definable. Indeed, if $\varphi(x, T)$ is the positive in T existential first-order formula

$$A(x) \vee (\exists y)(\exists z)(T(y) \wedge T(z) \wedge R(x, y, z)),$$

then PATH SYSTEMS is definable by the least fixed-point $\varphi^\infty(x)$ of $\psi(x, T)$. ∎

Example 6.8: ACYCLICITY

Let $\psi(x, S)$ be the positive in S universal first-order formula

$$(\forall y)(E(y, x) \rightarrow S(y)).$$

Clearly, the first stage $\psi^1(x)$ defines the set of all nodes x of in-degree equal to 0. Similarly, the second stage $\psi^2(x)$ defines the set of all nodes x that either have in-degree equal to 0 or have the property that if y is a node such that $E(y, x)$, then y has in-degree equal to 0. By continuing this analysis for all stages $\psi^n(x)$, $n \geq 1$, it can be seen that on every finite graph $\mathbf{G} = (V, E)$, the least fixed-point $\psi^\infty(x)$ defines the set of all nodes in V such that "no path down from x leads to a cycle", that is, the set of all nodes x such that there is no sequence of nodes $y_1, \ldots, y_m$ such that $E(x, y_1)$, $E(y_1, y_2)$, …, $E(y_{m-1}, y_m)$ and y_m is a node on a cycle of $\mathbf{G}$. It follows that ACYCLICITY is a LFP_1-definable query, since it is definable by the LFP_1-formula $(\forall x)\psi^\infty(x)$.

Although our main focus is on finite structures, it is worth pointing out that on every graph $\mathbf{G} = (V, E)$ (finite or infinite), the least fixed-point $\psi^\infty(x)$ of $\psi(x, S)$ defines the *well-founded part* of E, that is the set of all nodes x in V such that there is no infinite descending E-chain $E(x, y_1)$, $E(y_m, y_{m+1})$, $m \geq 1$. For finite graphs, of course, the well-founded part of E is the set of all nodes x such that no path down from x leads to cycle. It should also be pointed out that the closure ordinal of the formula $\psi(x, S)$ can be arbitrarily large. Indeed, if $\mathbf{G} = (V, E)$ is a well-ordering of rank α, then $\mathrm{cl}(\psi) = \alpha$. ∎

Example 6.9: GEOGRAPHY

Every finite graph $\mathbf{G} = (V, E)$ gives rise to a two-person game played according to the following rules: Player I and Player II take turns picking nodes in V; if a is the last node picked, then the player whose turn is to play next must pick a node b such that $E(a, b)$, else this player loses. This abstracts a game played between two children in which they take turns and write down the name of a city whose first letter is the same as the last letter of the city written down in the previous step of the game.

Consider now the following unary query, called GEOGRAPHY: given a finite graph $\mathbf{G} = (V, E)$ and a node v in V, find the set of all nodes v that are winning positions for Player I. It is well known that this query is P-complete (see [GHR95]); moreover, using Ehrenfeucht-Fraïssé-games, it can be shown that it is not FO-definable. It is easy to see, however, that GEOGRAPHY is LFP_1-definable. Indeed, if $\varphi(x, S)$ is the positive in S universal-existential first-order formula

$$(\forall y)\neg E(x, y) \vee (\forall y)(E(x, y) \rightarrow (\exists z)(E(y, z) \wedge S(z))),$$

then on every graph $\mathbf{G} = (V, E)$, the least fixed-point $\varphi^{\infty}(x)$ of $\varphi(x, S)$ defines the set of all winning positions for Player I. ∎

As a byproduct of Theorem 6.2 and the preceding Examples 6.7 and 6.9, we can determine the data complexity of LFP and of LFP_1.

Proposition 6.10: *The data complexity of* LFP *is* P*-complete; the data complexity of* LFP_1 *is* P*-complete as well.*

Proof: *(Sketch)* Every LFP-definable query is in P, because, given a finite σ-structure, the least fixed-points and greatest fixed-points of LFP-formulas can be evaluated by iterating the stages of the associated operator a polynomial number of times in the size of the given structure (and each step in this iteration can be carried out in time bounded by a polynomial in the size of the structure). Thus, the data complexity of LFP (and, a fortiori, of LFP_1) is in P. Since LFP_1 can express P-complete queries, such as PATH SYSTEM and GEOGRAPHY, it follows that the data complexity of LFP_1 (and, a fortiori, of LFP) is P-complete. ∎

It is also known that the expression complexity and the combined complexity of LFP and of LFP_1 are EXPTIME-complete [Var82]; this is yet another instance of the exponential-gap phenomenon between data complexity and the expression (and combined) complexity of a logic.

Let σ be a vocabulary containing at least one relation symbol of arity 2 or higher, and let $\mathcal{F}$ be the class of all finite σ-structures. Although LFP can express P-complete queries on $\mathcal{F}$, it cannot express every polynomial-time computable query on $\mathcal{F}$. Indeed, in the next section we will show that LFP cannot express *counting* queries, such as EVEN CARDINALITY. Thus, the following proper containments hold on $\mathcal{F}$:

$$\mathrm{FO}(\mathcal{F}) \subset \mathrm{LFP}(\mathcal{F}) \subset \mathrm{P}.$$

Immerman [Imm82, Imm86] and Vardi [Var82], however, showed that LFP can express all polynomial-time computable queries on classes of *ordered* finite structures, that is, on classes of finite structures in which one of the relations is a linear order on the universe of the structure.

Theorem 6.11: [Immerman [Imm82, Imm86], Vardi [Var82]] . *Let $\mathcal{C}$ be a class of ordered finite structures. The following are equivalent for a query Q on $\mathcal{C}$.*

- *Q is polynomial-time computable.*
- *Q is* LFP*-definable on $\mathcal{C}$.*

In other words, $\mathrm{P}(\mathcal{C}) = \mathrm{LFP}(\mathcal{C})$.

So far, we have focused on recursive specifications of single queries. In many areas of computer science, however, it is quite common to specify objects recursively using *mutual recursion*, that is, the object of interest is defined together with several other auxiliary objects via a simultaneous recursive specification. In what follows, we formalize the mechanism of mutual recursion for queries and explore its basic properties.

Definition 6.12: Let A be a set.

- A *system* of operators on a A is a finite sequence $(\Phi_1, \ldots, \Phi_m)$ of mappings

$$\Phi_i : \mathcal{P}(A^{k_1}) \times \cdots \times \mathcal{P}(A^{k_m}) \to \mathcal{P}(A^{k_i}), \quad 1 \leq i \leq m.$$

- A sequence $(P_1, \ldots, P_m)$ of relations on A is a *fixed-point* of the system $(\Phi_1, \ldots, \Phi_m)$ if $P_i \subseteq A^{k_i}$, for $1 \leq i \leq m$, and $(\Phi_1(P_1), \ldots, \Phi_m(P_m)) = (P_1, \ldots, P_m)$.

- A sequence $(P_1, \ldots, P_m)$ of relations on A is the *least fixed-point* of the system $(\Phi_1, \ldots, \Phi_m)$ if it is a fixed-point of $(\Phi_1, \ldots, \Phi_m)$ and for every fixed-point $(P'_1, \ldots, P'_m)$ of $(\Phi_1, \ldots, \Phi_m)$, we have that $P_i \subseteq P'_i$, for $1 \leq i \leq m$.

  We write $\mathbf{lfp}(\Phi_1, \ldots, \Phi_m)$ to denote the least fixed-point of $(\Phi_1, \ldots, \Phi_m)$, if it exists.

- A system $(\Phi_1, \ldots, \Phi_m)$ is *monotone* if for every two sequences $(P_1, \ldots, P_m)$, $(P'_1, \ldots, P'_m)$ of relations on A such that $P_i \subseteq P'_i \subseteq A^{k_i}$, $1 \leq i \leq m$, we have that $\Phi_i(P_1, \ldots, P_m) \subseteq \Phi_i(P'_1, \ldots, P'_m)$, for $1 \leq i \leq m$.

- The (finite) *stages* $(\Phi_1^n, \ldots, \Phi_m^n)$, $n \geq 1$, of the system $(\Phi_1, \ldots, \Phi_m)$ are defined by the following simultaneous induction:

$$\left| \begin{array}{lcll} \Phi_i^1 & = & \Phi_i(\emptyset, \ldots, \emptyset), & 1 \leq i \leq m \\ \Phi_i^{n+1} & = & \Phi_i(\Phi_1^n, \ldots, \Phi_m^n), & 1 \leq i \leq m. \end{array} \right.$$

  In general, for every ordinal α, the *stage* $(\Phi_1^\alpha, \ldots, \Phi_m^\alpha)$ is defined by the simultaneous transfinite induction

$$\Phi_i^\alpha \quad = \quad \Phi_i(\textstyle\bigcup_{\beta<\alpha} \Phi_1^\beta, \ldots, \bigcup_{\beta<\alpha} \Phi_m^\beta), \quad 1 \leq i \leq m.$$

  We put $(\Phi_1^\infty, \ldots, \Phi_m^\infty) = (\bigcup_\alpha \Phi_1^\alpha, \ldots, \bigcup_\alpha \Phi_m^\alpha)$ for the union of the stages.

Using simultaneous transfinite induction, it is easy to verify that the Knaster-Tarski Theorem 6.2 extends to monotone systems of operators.

Theorem 6.13: *Let A be a set and $(\Phi_1, \ldots, \Phi_m)$ a monotone system of operators on A.*

- *$(\Phi_1, \ldots, \Phi_m)$ has a least fixed-point $\mathbf{lfp}(\Phi_1, \ldots, \Phi_m)$.*

- *There is an ordinal γ such that*

$$\mathbf{lfp}(\Phi_1, \ldots, \Phi_m) = (\Phi_1^\infty, \ldots, \Phi_m^\infty) = (\Phi_1^\gamma, \ldots, \Phi_m^\gamma) = (\Phi_1^\delta, \ldots, \Phi_m^\delta), \text{ for every } \delta > \gamma.$$

  If A is a finite set, then there is an integer $s \leq \prod_{i=1}^m |A|^{k_i}$ such that

$$\mathbf{lfp}(\Phi_1, \ldots, \Phi_m) = (\Phi_1^\infty, \ldots, \Phi_m^\infty) = (\Phi_1^s, \ldots, \Phi_m^s) = (\Phi_1^\delta, \ldots, \Phi_m^\delta), \text{ for every } \delta > s.$$

- *The least fixed-point* $\mathbf{lfp}(\Phi_1, \ldots, \Phi_m)$ *of* $(\Phi_1, \ldots, \Phi_m)$ *is equal to the (coordinatewise) intersection of all fixed-points of* $(\Phi_1, \ldots, \Phi_m)$.

Definition 6.14: Let $(\Phi_1, \ldots, \Phi_m)$ be a monotone system of operators on A. The *closure ordinal* of this system, denoted by $\mathrm{cl}(\Phi_1, \ldots, \Phi_m)$, is the smallest ordinal ordinal γ such that

$$(\Phi_1^\gamma, \ldots, \Phi_m^\gamma) = (\bigcup_{\beta<\gamma} \Phi_1^\beta, \ldots, \bigcup_{\beta<\gamma} \Phi_m^\beta).$$

We now consider systems of operators arising from first-order formulas.

Definition 6.15: Let σ be a vocabulary.

- A *system* of first-order formulas is a sequence

  $$(\varphi_1(\mathbf{x}_1, S_1, \ldots, S_m), \ldots, \varphi_m(\mathbf{x}_m, S_1, \ldots, S_m)),$$

  of first-order formulas over the vocabulary $\sigma \cup \{S_1, \ldots, S_m)$ such that each $\mathbf{x}_i$ is a sequence of variables whose length is equal to the arity of the relation symbol S_i, $1 \leq i \leq m$ (of course, some of the relation symbols $S_1, \ldots, S_m$ may not occur in the formula φ_i).

- If $\mathbf{A}$ is a σ-structure, then a system of first-order formulas as above gives rise to a system $(\Phi_1, \ldots, \Phi_m)$ of operators on A such that for every $i \leq m$,

  $$\Phi_i(P_1, \ldots, P_m) = \{\mathbf{a}_i : \mathbf{A} \models \varphi_i(\mathbf{a}_i, P_1, \ldots, P_m)\}.$$

- Let $(\varphi_1(\mathbf{x}_1, S_1, \ldots, S_m), \ldots, \varphi_m(\mathbf{x}_m, S_1, \ldots, S_m))$ be a system of first-order formulas each of which is positive in $S_1, \ldots, S_m$. We write $(\varphi_1^\infty, \ldots, \varphi_m^\infty)$ for the least fixed-point of the monotone system associated with this system of positive first-order formulas. Similarly, we write $\mathrm{cl}(\varphi_1, \ldots, \varphi_m)$ for the closure ordinal of this system.

Example 6.16: Even Path and Odd Path.

Let $\varphi_1(x, y, S_1, S_2)$ be the positive first-order formula

$$E(x, y) \vee (\exists z)(E(x, z) \wedge S_2(z, y))$$

and let $\varphi_2(x, y, S_1, S_2)$ be the positive first-order formula

$$(\exists z)(E(x, z) \wedge S_1(z, y)).$$

Consider the least fixed-point $(\varphi_1^\infty, \varphi_2^\infty)$ of the system consisting of these two formulas. It is easy to see that φ_1^∞ defines the Odd Path query OP on graphs and φ_2^∞ defines the Even Path query EP on graphs, where for every graph $\mathbf{G} = (V, E)$

$$\begin{aligned} OP(\mathbf{G}) &= \{(a, b) \in V^2 : \text{there is a path of odd length from } a \text{ to } b\} \\ EP(\mathbf{G}) &= \{(a, b) \in V^2 : \text{there is a path of even length from } a \text{ to } b\}. \end{aligned}$$

∎

The next result asserts that least fixed-points of systems of positive first-order formulas have the same expressive power as LFP_1-formulas. Moreover, it asserts that systems consisting of positive existential and positive universal first-order formulas are as powerful as systems of arbitrary positive first-order formulas.

Theorem 6.17: *Let σ be a vocabulary, $\mathcal{C}$ a class of σ-structures each of which has at least two elements in its universe, and Q a query on $\mathcal{C}$. Then the following statements are equivalent:*

1. *Q is LFP_1-definable on $\mathcal{C}$.*
2. *There is system $(\varphi_1(\mathbf{x}_1, S_1, \ldots, S_m), \ldots, \varphi_m(\mathbf{x}_m, S_1, \ldots, S_m))$ of positive first-order formulas such that φ_m^∞ defines Q on $\mathcal{C}$ and each $\varphi_i(\mathbf{x}_i, S_1, \ldots, S_m)$ is either a positive existential first-order formula or a positive universal first-order formula, $1 \leq i \leq m$.*
3. *There is system $(\varphi_1(\mathbf{x}_1, S_1, \ldots, S_m), \ldots, \varphi_m(\mathbf{x}_m, S_1, \ldots, S_m))$ of positive first-order formulas such that φ_m^∞ defines Q on $\mathcal{C}$.*

Proof: *(Sketch)* Since the direction (2) $\Rightarrow$ (3) is trivial, it suffices to establish the directions (1) $\Rightarrow$ (2) and (3) $\Rightarrow$ (1). The proof of (1) $\Rightarrow$ (2) is by induction on the construction of LFP_1-formulas. For concreteness, suppose that we are given the LFP_1-formula $(\exists y)\varphi^\infty(x, y)$, where $\varphi(x, y, S)$ is a positive in S first-order formula of the form $(\forall z)(\exists w)\theta(x, y, z, w, S)$ with $\theta(x, y, z, w, S)$ a quantifier-free formula and S a binary relation symbol. Consider the system

$$(\varphi_1(x, y, z, S_1, S_2, S_3), \varphi_2(x, y, S_1, S_2, S_3), \varphi_3(x, S_1, S_2, S_3),$$

where

$$\begin{aligned} \varphi_1(x, y, z, S_1, S_2, S_3) &\equiv (\exists w)\theta(x, y, z, w, S_2) \\ \varphi_2(x, y, S_1, S_2, S_3) &\equiv (\forall z)S_1(x, y, z) \\ \varphi_3(x, S_1, S_2, S_3) &\equiv (\exists y)S_2(x, y) \end{aligned}$$

with S_1 a ternary relation symbol, S_2 a binary one, and S_3 a unary one. By transfinite induction on the stages and using the monotonicity of the formulas, it is not hard to verify that the given LFP_1-formula $(\exists y)\varphi^\infty(x, y)$ is logically equivalent to $\varphi_3^\infty(x)$.

The other steps of this direction are quite similar. For instance, suppose we are given the LFP_1-formula $\varphi^\infty(x, y) \wedge \psi^\infty(x, y)$, where $\varphi(x, y, S)$ and $\psi(x, y, S)$ are positive in S first-order formulas. By induction hypothesis, we may assume that there are systems $(\varphi_1, \ldots, \varphi_m)$ and $(\psi_1, \ldots, \psi_s)$ of positive existential and positive universal first-order formulas such that $\varphi^\infty(x, y)$ is logically equivalent to $\varphi_m^\infty(x, y)$ and $\psi^\infty(x, y)$ is logically equivalent to $\psi_s^\infty(x, y)$. Suppose that the relation variables in the first system are $S_1, \ldots, S_m$ and in the second system $T_1, \ldots, T_s$. Consider the system

$$(\varphi_1, \ldots, \varphi_m, \psi_1, \ldots, \psi_s, \chi),$$

where χ is the formula $S_m(x, y) \wedge T_s(x, y)$. Then the given LFP_1-formula $\varphi^\infty(x, y) \wedge \psi^\infty(x, y)$ is logically equivalent to $\chi^\infty(x, y)$.

We now focus on the direction (3) $\Rightarrow$ (1). Again for concreteness, suppose we are given the system $(\varphi_1(x, S_1, S_2), \varphi_2(y, z, S_1, S_2))$, where φ_1 and φ_2 are positive in S_1, S_2 first-order formulas, S_1 is a unary relation symbol, and S_2 is a binary relation symbol. Let S be a 5-ary relation symbol and let $\varphi(u, v, x, y, z, S)$ be the positive in S first-order formula

$$((u \neq v) \wedge \varphi_1(x, \{x' : (\exists u')(\exists v')(u' \neq v') \wedge S(u', v', x'u', u')\}, \{(y', z') : (\exists u')S(u', u', u', y', z')\}))$$

$$\vee((u = v) \wedge \varphi_2(y, z, \{x' : (\exists u')(\exists v')(u' \neq v') \wedge S(u', v', x'u', u')\}, \{(y', z') : (\exists u')S(u', u', u', y', z')\})).$$

By induction on the stages and using the monotonicity of the formulas, one can verify that for every ordinal α, we have that $\varphi_1^\alpha(x)$ is logically equivalent to $(\exists u)(\exists v)((u \neq v) \wedge \varphi_1^\alpha(u, v, x, u, u))$, while

at the same time $\varphi_2^\alpha(y,z)$ is logically equivalent to $(\exists u)(\varphi^\alpha(u,u,u,y,z)$. It follows that $\varphi_1^\infty(x)$ is logically equivalent to $(\exists u)(\exists v)((u \neq v) \wedge \varphi^\infty(u,v,x,u,u))$ and $\varphi_2^\infty(y,z)$ is logically equivalent to $(\exists u)\varphi^\infty(u,u,u,y,z)$. ∎

Several remarks are in order now. In Moschovakis' book [Mos74], the equivalence between statements (1) and (2) is attributed to P. Aczel (unpublished result); the same monograph contains a detailed proof of the direction (3) ⇒ (1), which is often called the Simultaneous Induction Lemma.

The preceding Theorem 6.17 is a basic and extremely useful result about the expressive power of least fixed-point logic LFP_1. It shows that, although on the face of Definition 6.5 the syntax of LFP_1 is quite restricted, LFP_1 is robust enough to simulate least fixed-points of systems of positive first-order formulas. It also facilitates the task of showing that a query is LFP_1-definable, because quite often it is easier to define a query by mutual recursion using a system of positive first-order formulas. Furthermore, the equivalence between statements (1) and (2) in Theorem 6.17 reveals that no hierarchy of progressively more expressive sublogics of LFP_1 arises by restricting the length of quantifier alternation in the formulas occurring in systems. Thus, there are just two main sublogics of least fixed-point obtained by imposing restrictions on the quantification pattern: ELFP_1 and ULFP_1. The former is the sublogic of LFP_1 determined by systems of positive existential first-order formulas, while the latter is the sublogic of LFP_1 determined by systems of positive universal first-order formulas. In what follows, we will consider certain fragments of ELFP_1 that have played an important role in database theory.

6.3 Datalog and Datalog($\neq$)

Datalog can be succinctly described as the data sublanguage of logic programming. More formally, a *Datalog program* π is a finite set of function-free, $\neq$-free, and negation-free *rules* of the form:

$$t_0 \ :- \ t_1, \ldots, t_m$$

where each t_i is an atomic formula $R(x_1, \ldots, x_n)$ for some n-ary relation symbol, $n \geq 1$; in addition, t_0 may be a 0-ary relation symbol standing for "true". The expression t_0 is the *the head* of the rule, while the expression $t_1, \ldots, t_m$ is the *body* of the rule. The relation symbols that occur in the heads of the rules of a given Datalog program π are usually called the *intensional database* predicates (IDBs) of π, while all others are the *extensional database* predicates (EDBs) of π. One of the IDBs is designated as the *goal* of π. Note that IDBs may occur in the bodies of rules and, thus, a Datalog program can be viewed as a simultaneous recursive specification of the IDBs. Given a set of relations for the EDBs of π, each IDB is originally instantiated to the empty relation and then the rules of the Datalog program are applied repeatedly until no new tuples are added to the IDBs. An application of each rule entails adding to the IDB in the head of each rule all tuples that satisfy the head of the rule. This is an informal description of the "bottom-up" evaluation of a Datalog program and it provides the *procedural semantics* of that program. Alternatively, a Datalog program can be given *declarative semantics* using least fixed-points of recursive specification (see [Ull89, AHV95] for precise definitions). The query definable by a Datalog program π is the query whose value on a structure $\mathbf{A}$ is the value of the goal of π with the relations of $\mathbf{A}$ as EDBs of π. If the goal of π is 0-ary, then π defines a Boolean query.

Example 6.18: Transitive Closure Revisited

Consider the following Datalog program having E as its only EDB and S as its only IDB:

$$\begin{array}{|lcl} S(x,y) & :- & E(x,y) \\ S(x,y) & :- & E(x,z), S(z,y) \end{array}$$

This program defines the TRANSITIVE CLOSURE query. Note that the TRANSITIVE CLOSURE query is also definable by the following Datalog program:

$$\left|\begin{array}{lll} S(x,y) & :- & E(x,y) \\ S(x,y) & :- & S(x,z), S(z,y) \end{array}\right.$$

∎

Example 6.19: PATH SYSTEMS Revisited
Consider the following Datalog program having A and R as its EDBs and T as its only IDB.

$$\left|\begin{array}{lll} T(x) & :- & A(x) \\ T(x) & :- & T(y), T(z), R(x,y,z) \end{array}\right.$$

This program defines the PATH SYSTEMS query. ∎

Note that the preceding Example 6.19 reveals that the data complexity of Datalog is P-complete, that is, it is the same as the full LFP, even though Datalog is a small fragment of it.

Example 6.20: NON-2-COLORABILITY
Consider the following Datalog program having E as its only EDB, O and Q as its IDBs, and Q as its 0-ary goal predicate.

$$\left|\begin{array}{lll} O(x,y) & :- & E(x,y) \\ O(x,y) & :- & E(x,z), E(z,w), O(z,y) \\ Q & :- & O(x,x) \end{array}\right.$$

In this program, O defines the set of pairs of nodes connected via a path of odd length. Consequently, Q defines the set of all graphs that contain a cycle of odd length, that is, the set of all graphs that are not 2-colorable. ∎

As seen earlier in Example 6.6, the TRANSITIVE CLOSURE query is definable by a positive in S existential first-order formula. Similarly, as seen in Example 6.7, the PATH SYSTEMS query is definable by a positive in T existential first-order formula. Moreover, these formulas are $\neq$-free and negation-free (that is, they are also positive in E). In the other direction, the NON-2-COLORABILITY query is definable by the formula $(\exists x)\varphi^\infty(x,x)$, where $\varphi^\infty(x,y,O)$ is the following existential first-order formula that is positive in O and E, and also $\neq$-free:

$$E(x,y) \vee (\exists z)(E(x,z) \wedge E(z,w) \wedge O(z,y)).$$

Chandra and Harel [CH85] were the first to point out that these connections are not accidental.

Proposition 6.21: [CH85] *Let $\mathcal{C}$ be a class of structures and Q a query on $\mathcal{C}$. Then the following statements are equivalent:*

- *Q is definable on $\mathcal{C}$ by a Datalog program.*
- *Q is definable on $\mathcal{C}$ by φ_m^∞ for some system $(\varphi_1, \ldots, \varphi_m)$ of first-order formulas such that each φ_i is of the form $(\exists \mathbf{z}_i)\psi_i$ and ψ_i is a conjunction of atomic formulas.*

Proof: *(Hint)* Every rule of a Datalog program gives rise to a formula of a system. The body of the rule is first rewritten as a conjunction of the atomic relations occurring in it; after this, the variables occurring in the body, but not in the head of the rule, are existentially quantified out. Conversely, every formula in a system in (2) can be viewed as a rule of a Datalog program. ∎

Although Datalog can express P-complete queries, it is strictly less expressive than LFP_1. As we will see next, some of the limitations of Datalog are consequences of *preservation* properties possessed by Datalog queries.

Definition 6.22: Let σ be a vocabulary.

- A *homomorphism* $h : \mathbf{A} \to \mathbf{B}$ between two σ-structures $\mathbf{A}$ and $\mathbf{B}$ is a mapping h from the universe A of $\mathbf{A}$ to the universe B of $\mathbf{B}$ with the following properties:
  - For every constant symbol c in σ, we have that $h(c^{\mathbf{A}}) = c^{\mathbf{B}}$;
  - For every relation symbol R in σ and every tuple $\mathbf{a}$ from A, if $\mathbf{a} \in R^{\mathbf{A}}$, then $h(\mathbf{a}) \in R^{\mathbf{B}}$.
- Let Q be a k-ary query on a class $\mathcal{C}$ of σ-structures. We say that Q is *preserved under homomorphisms* if for every two structures $\mathbf{A}$, B in $\mathcal{C}$, every homomorphism $h : \mathbf{A} \to \mathbf{B}$, and every k-tuple $\mathbf{a}$ from A, if $\mathbf{a} \in Q(\mathbf{A}$, then $h(\mathbf{a}) \in Q(\mathbf{B})$.
- Let Q be a Boolean query on a class $\mathcal{C}$ of σ-structures. We say that Q is *preserved under homomorphisms* if for every two structures $\mathbf{A}$, $\mathbf{B}$ in $\mathcal{C}$ such that there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$, if $\mathbf{A} \models Q$, then $\mathbf{B} \models Q$.

Proposition 6.23: *Let σ be a vocabulary. Every Datalog-definable query is preserved under homomorphisms on the class $\mathcal{S}$ of all σ-structures.*

Proof: *(Sketch)* The preceding Proposition 6.21 implies that the system of operators associated with a Datalog program is definable by first-order formulas that are positive in every relation symbol occurring in them and also are $\neq$-free. Using this fact and induction on the stages of the system, it is easy to show that each stage of the system is preserved under homomorphisms on $\mathcal{S}$. ∎

Consider the existential first-order sentence $(\exists x)(\exists y)(x \neq y)$ asserting that there are at least two distinct elements in the universe. An immediate consequence of Proposition 6.23 is that this sentence is *not* equivalent to any Datalog sentence, because it is not preserved homomorphisms.

Datalog($\neq$) is the extension of Datalog in which $\neq$ are allowed in the rules. The next example illustrates the syntax of Datalog($\neq$).

Example 6.24: Node-Avoiding Path:

Let Q be the following query on graphs: given a graph $\mathbf{G} = (V, E)$ and three nodes a, b, c, is there a path from a to c that avoids w?

This query is definable by the following Datalog($\neq$)-program.

$$\begin{aligned} T(x, y, w) &:- E(x, y) \wedge w \neq x \wedge w \neq y \\ T(x, y, w) &:- E(x, z) \wedge T(z, y, w) \wedge w \neq x. \end{aligned}$$

It is easy to see that Q is not preserved under homomorphisms and, consequently, it is not expressible in Datalog. ∎

Note that the above query is also definable by the least fixed-point of the positive in T existential first-order formula

$$(E(x,y) \wedge w \neq x \wedge w \neq y) \vee (\exists z)(E(x,z) \wedge T(z,y,w) \wedge w \neq x).$$

This is an instance of a more general result that is analogous to Proposition 6.21. Specifically, a query Q is definable on a class $\mathcal{C}$ by a Datalog($\neq$)-program if and only if Q is definable on $\mathcal{C}$ by φ_m^∞ for some system $(\varphi_1, \ldots, \varphi_m)$ of first-order formulas such that each φ_i is of the form $(\exists \mathbf{z}_i \psi_i$ and ψ is a conjunction of atomic formulas and inequalities $\neq$.

We now present an example of a query on undirected graphs that is definable by a Datalog($\neq$)-program, but proving this fact requires some machinery from graph theory.

Example 6.25: The Even Simple Path query asks: given a graph $\mathbf{G} = (V, E)$ and two nodes a, b, is there a simple path of even length from a to b?

Using results of Fortune, Hopcroft and Wiley [FHW80] about the Graph Homeomorphism Problem, it can be shown that Even Simple Path on directed graphs is an NP-complete problem. In contrast, there is a polynomial-time algorithm for Even Simple Path when the inputs are undirected graphs. Moreover, in an unpublished note, Yannakakis showed that the following Datalog($\neq$)-program with Q as its goal defines the Even Simple Path query on undirected graphs.

$$\begin{array}{lll}
T(x,y,w) & :- & E(x,y) \wedge w \neq x \wedge w \neq y \\
T(x,y,w) & :- & E(x,z) \wedge T(z,y,w) \wedge w \neq x \\
P(x,y) & :- & E(x,y) \\
P(x,y) & :- & Q(x,w), E(w,y), T(x,w,y) \\
Q(x,y) & :- & P(x,w), P(w,y), T(x,w,y).
\end{array}$$

The correctness of this program is established by proving that on undirected graphs:

- T defines the Node-Avoiding Path query.
- P defines the Odd Simple Path query (that is, "is there a simple path of odd length from a to b?").
- Q defines the Even Simple Path query.

The proof proceeds by induction on the stages of the above Datalog($\neq$)-program and makes use of Menger's Theorem, a well-known result in graph theory which asserts that if an undirected graph $G = (V, E)$ and two nodes a, b have the property that every two paths from a to b intersect at some intermediate node, then there is a node c different from a and b such that all paths from a to b intersect at c (Menger's Theorem is a special case of the Max Flow-Min Cut Theorem, see [Die97]). ∎

A *one-to-one homomorphism* between two σ-structures $\mathbf{A}$ and $\mathbf{B}$ is a homomorphism $h : \mathbf{A} \to \mathbf{B}$ that is also a one-to-one mapping from A to B. The next result is proved along the lines of the proof of Proposition 6.23.

Proposition 6.26: *Every Datalog($\neq$)-definable query is preserved under one-to-one homomorphisms on the class $\mathcal{S}$ of all σ-structures.*

Consider the universal first-order sentence $(\forall x)(\forall y)(x \neq y \rightarrow E(x,y))$, which asserts that $\mathbf{G} = (V,E)$ is a complete graph. Since this sentence is not preserved under one-to-one homomorphisms, it is not equivalent to any Datalog($\neq$) sentence. Thus, on the class $\mathcal{G}$ of all finite graphs the Datalog($\neq$) is strictly more expressive than Datalog, but strictly less expressive than LFP_1.

Another difference between Datalog($\neq$) and LFP_1 has to do with closure ordinals on infinite structures. As seen earlier in Example 6.8, there are positive universal first-order formulas whose closure ordinal can be arbitrarily large on infinite structures. In contrast, it is not hard to prove that on every infinite structure the closure ordinal of every of Datalog($\neq$)-program is at most ω. Indeed, this follows from the fact that existential quantification distributes over an infinite union, that is, $(\exists x)(\bigcup_{n=1}^{\infty} P_n)$ is logically equivalent to $\bigcup_{n=1}^{\infty}(\exists x)P_n$.

6.4 The Complementation Problem for LFP_1 and a Normal Form for LFP

The *structure of arithmetic* is the structure $\mathbf{N} = (N,+,\times)$, where N is the set of all natural numbers, and $+$ and $\times$ are ternary relations for the graphs of the addition and multiplication functions on the natural numbers. The expressive power of LFP_1 on $\mathbf{N} = (N,+,\times)$ was first studied by Kleene [Kle55] and Spector [Spe61], who established the following important result, known as the Kleene-Spector Theorem (see [Mos74]).

Theorem 6.27: *Let $\mathbf{N} = (N,+,\times)$ be the stucture of arithmetic.*

- $\mathrm{LFP}_1(\mathbf{N}) = \mathrm{USO}(\mathbf{N})$*, that is, a relation $R \subseteq N^k$ is LFP_1-definable on $\mathbf{N}$ if and only if it is definable on $\mathbf{N}$ by a universal second-order formula.*
- $\mathrm{LFP}_1(\mathbf{N})$ *is not closed under complements.*

Several remarks are now in order, so that the Kleene-Spector Theorem be put into the right perspective. First, if σ is a vocabulary and $\mathbf{A}$ is an arbitrary σ-structure, then $\mathrm{LFP}_1(\mathbf{A}) \subseteq \mathrm{USO}(\mathbf{A}$. The reason for this is that if $\varphi(\mathbf{x},S)$ is a positive in S first-order formula over the vocabulary $\sigma \cup \{S\}$, then the least fixed-point $\varphi^{\infty}(\mathbf{x})$ is definable on $\mathbf{A}$ by the USO-formula

$$(\forall S)((\forall \mathbf{z})(\varphi(\mathbf{z},S) \leftrightarrow S(\mathbf{z}))) \rightarrow S(\mathbf{x})).$$

Indeed, the above USO-formula defines the least fixed-point of $\varphi(\mathbf{x},S)$, because, as seen in Theorem 6.2, the least fixed-point of a monotone operator is the intersection of all its fixed-points. If $\mathbf{A}$ is an arbitrary infinite σ-structure, then $\mathrm{LFP}_1(\mathbf{A})$ may be properly contained in $\mathrm{USO}(\mathbf{A})$; for instance, this is the case for the structure $\mathbf{Q} = (Q,<)$, where Q is the set of rational numbers and $<$ is the standard linear order on Q. In contrast, the Kleene-Spector Theorem asserts that the LFP_1-definable relation coincide with the USO-definable ones on the structure $\mathbf{N} = (N,+,\times)$ of arithmetic; thus, this result provides a "constructive" characterization of universal second-order logic on $\mathbf{N}$. Moschovakis [Mos74] has shown that the Kleene-Spector Theorem actually extends to countable structures $\mathbf{A}$ possessing a first-order *coding machinery* for finite sequences, that is, countable structures in which finite sequences of arbitrary length can be encoded by individual elements and decoded in a first-order definable way. Moreover, on such countable structures $\mathbf{A}$ there is a binary USO-definable relation whose projections are exactly all unary USO-definable relations (such relations are called *universal* USO-definable relations). Using this fact and a diagonalization argument, it can be shown that the USO-definable relations on such structures $\mathbf{A}$ are not closed under complements. In particular, the LFP_1-definable relations on $\mathbf{N}$ are not closed under complements.

Chandra and Harel [CH80] initiated the study of LFP on finite structures; moreover, motivated by the Kleene-Spector Theorem, they conjectured that the LFP_1-definable queries on the class $\mathcal{G}$ of all finite graphs are *not* closed under complements. This conjecture, however, was refuted by Immerman [Imm82, Imm86], who showed that if $\mathcal{C}$ is an arbitrary class of finite structures, then $\mathrm{LFP}_1\mathcal{C})$ is closed under complements. In what follows, we will outline a proof of this result and, in the process of doing so, we will present some other fundamental properties of LFP_1.

Definition 6.28: Let σ be a vocabulary.

- Let $\varphi(x_1, \ldots, x_k, S)$ a positive in S first-order formula over the vocabulary $\sigma \cup \{S\}$. For every σ-structure $\mathbf{A}$ and every k-tuple $\mathbf{a} \in A^k$, we put

$$|\mathbf{a}|_\varphi = \begin{cases} \min\{\alpha : \mathbf{A} \models \varphi^\alpha(\mathbf{a})\} & \text{if } \mathbf{A} \models \varphi^\infty(\mathbf{a}) \\ \infty & \text{if } \mathbf{A} \models \neg\varphi^\infty(\mathbf{a}) \end{cases}$$

- Let $\varphi(\mathbf{x}, S)$ be a positive in S first-order formula over $\sigma \cup \{S\}$ and let $\psi(\mathbf{y}, T)$ be a positive in T first-order formula over $\sigma \cup \{T\}$. The *stage comparison queries* $\preceq^*_{\varphi,\psi}$ and $\prec^*_{\varphi,\psi}$ associated with the formulas $\varphi(\mathbf{x}, S)$ and $\psi(\mathbf{y}, T)$ are the queries such that for every σ-structure $\mathbf{A}$,

$$\begin{aligned} \mathbf{a} \preceq^*_{\varphi,\psi} \mathbf{b} &\iff \varphi^\infty(\mathbf{a}) \wedge (|\mathbf{a}|_\varphi \leq |\mathbf{b}|_\psi) \\ \mathbf{a} \prec^*_{\varphi,\psi} \mathbf{b} &\iff |\mathbf{a}|_\varphi < |\mathbf{b}|_\psi \end{aligned}$$

  Note that if $\mathbf{a} \prec^*_{\varphi,\psi} \mathbf{b}$, then $|\mathbf{a}|_\varphi < \infty$ and, thus, $\mathbf{a} \in \varphi^\infty$.

- We write $\preceq^*_\varphi$ and $\prec_\varphi$ for the queries $\preceq^*_{\varphi,\varphi}$ and $\prec_{\varphi,\varphi}$, respectively.

The next two examples illustrate the meaning of the stage comparison queries $\preceq^*_\varphi$ and $\prec_\varphi$ for concrete formulas φ.

Example 6.29: Let $\mathbf{G} = (V, E)$ be a graph and let $\varphi(x, y, S)$ be the formula

$$E(x, y) \vee (\exists z)(E(x, z) \wedge S(z, y)),$$

whose least fixed-point defines the transitive closure of E. A moment's reflection reveals that $\preceq^*_\varphi$ is the *distance query* on graphs. More precisely, $(a, a') \preceq^*_\varphi (b, b')$ holds if and only if there is a path from a to a' and either there is no path from b to b' or the length of the shortest path from a to a' is at most equal to the length of the shortest path from b to b'. ∎

Example 6.30: As in Examples 6.7 and 6.19, assume that a proof system is encoded by a structure $\mathbf{S} = (F, A, R)$, where F is a set of formulas, A is a set of axioms, and R is a ternary rule of inference. Let $\psi(x, T)$ be the formula

$$A(x) \vee (\exists y)(\exists z)(T(y) \wedge T(z) \wedge R(x, y, z)),$$

whose least fixed-point defines the set of all theorems of this proof system. Then the stage comparison queries $\preceq^*_\psi$ and $\prec^*_\psi$ compare lengths of derivations of theorems of $\mathbf{S}$. In particular, $f \prec^*_\psi g$ holds if and only if f is a theorem of $\mathbf{S}$ and either g is not a theorem of $\mathbf{S}$ or f has a derivation in the proof system $\mathbf{S}$ that is shorter than any derivation of g. ∎

Theorem 6.31: [The Stage Comparison Theorem - Moschovakis [Mos74]] *Let σ be a vocabulary. If $\varphi(\mathbf{x}, S)$ and $\psi(\mathbf{y}, T)$ are a positive first-order formulas, then the stage comparison queries $\preceq^*_{\varphi,\psi}$ and $\prec^*_{\varphi,\psi}$ are LFP_1-definable on the class of all σ-structures.*

Proof: *(Hint:)* The stage comparison queries satisfy the equivalences:

$$\begin{array}{lll} \mathbf{x} \preceq^*_{\varphi,\psi} \mathbf{y} & \iff & \Phi^{|\mathbf{y}|_\psi}(\mathbf{x}) \iff \Phi(\mathbf{x}, \{\mathbf{x}' : |\mathbf{x}'|_\varphi < |\mathbf{y}|_\psi\}) \\ \mathbf{x} \prec^*_{\varphi,\psi} \mathbf{y} & \iff & \neg\Psi^{|\mathbf{x}|_\varphi}(\mathbf{y}) \iff \neg\Psi(\mathbf{y}, \{\mathbf{y}' : |\mathbf{y}'|_\psi < |\mathbf{x}|_\varphi\}). \end{array}$$

Note that if $\mathbf{x} \in \varphi^\infty$, then, for every $\mathbf{y}'$, we have that $|\mathbf{y}'|_\psi < |\mathbf{x}|_\varphi$ holds if and only if $\neg(\mathbf{x} \preceq^*_{\varphi,\psi} \mathbf{y}')$. It follows that the stage comparison queries satisfy the following recursive specifications:

$$\begin{array}{lll} \mathbf{x} \preceq^*_{\varphi,\psi} \mathbf{y} & \iff & \varphi(\mathbf{x}, \{\mathbf{x}' : \mathbf{x}' \prec^*_{\varphi,\psi} \mathbf{y}\}) \\ \mathbf{x} \prec_{\varphi,\psi} *\mathbf{y} & \iff & \neg\psi(\mathbf{y}, \{\mathbf{y}' : \neg\ (\mathbf{x} \preceq^*_{\varphi,\psi} \mathbf{y}'\}). \end{array}$$

This motivates considering the system $(\chi_1(x, y, S_1, S_2), \chi_2(x, y, S_1, S_2))$ of the first-order formulas

$$\begin{array}{lll} \chi_1(x, y, S_1, S_2) & \equiv & \varphi(\mathbf{x}, \{\mathbf{x}' : S_2(\mathbf{x}', \mathbf{y})\}) \\ \chi_2(x, y, S_1, S_2) & \equiv & \neg\psi(\mathbf{y}, \{\mathbf{y}' : \neg S_1(\mathbf{x}, \mathbf{y}')\}). \end{array}$$

Note that these formulas are positive in both S_1 and S_2, thus their system has a least fixed-point $(\chi_1^\infty, \chi_2^\infty)$. Using transfinite induction, it can be shown that χ_1^∞ defines $\preceq^*_{\varphi,\psi}$, and that χ_2^∞ defines $\prec^*_{\varphi,\psi}$. ∎

While the stage comparison theorem is a result about the class of all structures, the next theorem is rather special to classes of finite structures.

Theorem 6.32: [The Complementation Theorem for LFP_1 - Immerman [Imm82, Imm86]] *Let σ be a vocabulary. If $\mathcal{C}$ is a class of finite σ-structures, then $\mathrm{LFP}_1(\mathcal{C})$ is closed under complements.*

Proof: *(Sketch:)* It suffices to show that if $\varphi(\mathbf{x}, S)$ is a positive in S first-order formula over the vocabulary $\sigma \cup \{S\}$, then the complement $\neg\varphi^\infty$ is LFP_1-definable on $\mathcal{C}$.

Let Max_φ be the query that, given a σ-structure $\mathbf{A}$, returns the set of all tuples $\mathbf{a}$ in φ^∞ on $\mathbf{A}$ such that for every $\mathbf{b} \in \varphi^\infty$ on $\mathbf{A}$ we have that $|\mathbf{b}|_\varphi \leq |\mathbf{a}|_\varphi$. In other words, $\mathrm{Max}_\varphi(\mathbf{A})$ consists of all tuple from $\mathbf{A}$ that enter the "last" stage of the evaluation of φ^∞ on $\mathbf{A}$. Note that if $\mathbf{A}$ is an infinite structure, then $\mathrm{Max}_\varphi(\mathbf{A})$ may be empty, because there may be no "last" stage in the evaluation of φ (this happens precisely when the closure ordinal $\mathrm{cl}\varphi$ on $\mathbf{A}$ is a limit ordinal). For instance, this is the case when $\mathbf{G} = (V, E)$ is a graph of infinite diameter and $\varphi(x, y, S)$ is the formula whose least fixed-point defines the transitive closure of the edge relation E. In contrast, $\mathbf{A}$ is a finite structure, then $\mathrm{Max}_\varphi(\mathbf{A}) \neq \emptyset$ (unless $\varphi^\infty = \emptyset$ on $\mathbf{A}$).

We will now show that Max_φ is LFP_1-definable on the class of all finite σ-structures. Note that Max_φ satisfies the equivalence

$$\mathbf{a} \in \mathrm{Max}_\varphi(\mathbf{A}) \iff \mathbf{A} \models (\mathbf{a} \in \varphi^\infty) \wedge (\forall \mathbf{b})(|\mathbf{a}|_\varphi < |\mathbf{b}|_\varphi \to |\mathbf{a}|_\varphi + 1 < |\mathbf{b}|_\varphi).$$

It is easy to find a positive first-order formula ψ such that on finite structures ψ simulates φ with a "one-step" delay, that is, for every tuple $\mathbf{c} \in \varphi^\infty$, we have that $\mathbf{c}$ enters ψ^∞ exactly one stage after the stage it enters φ^∞. Using stage comparison queries, the above equivalence can be rewritten as

$$\mathbf{a} \in \mathrm{Max}_\varphi(\mathbf{A}) \iff \mathbf{A} \models (\mathbf{a} \in \varphi^\infty) \wedge (\forall \mathbf{b})((\mathbf{b} \preceq^*_\varphi \mathbf{a}) \vee (\mathbf{a} \prec^*_{\psi,\varphi} \mathbf{b})).$$

The Stage Comparison Theorem 6.31 immediately implies that Max_φ is LFP_1-definable on the class of all finite σ-structures.

It is now easy to show that the complement $\neg\varphi^\infty$ is LFP_1-definable on the class of all finite σ-structures. Indeed, if $\mathbf{A}$ is a finite σ-structure, then

$$\mathbf{A} \models \neg\varphi^\infty(\mathbf{a}) \Longleftrightarrow \mathbf{A} \models (\exists \mathbf{y})(\mathbf{y} \in \mathrm{Max}_\varphi \ \wedge \mathbf{y} \prec_\varphi^* \mathbf{a}). \blacksquare$$

With some extra work and using the ideas in the proof of Theorem 6.32, it is possible to establish the following *normal form* for least fixed-point logic LFP on classes of finite structures.

Theorem 6.33: [Imm82, Imm86] *If σ is a vocabulary and $\mathcal{C}$ is a class of finite σ-structures, then every* LFP*-definable query on $\mathcal{C}$ is* LFP_1*-definable on $\mathcal{C}$. Consequently,* $\mathrm{LFP}(\mathcal{C}) = \mathrm{LFP}_1(\mathcal{C})$.

Informally, this result asserts that on finite structures the nesting of least-fixed points, greatest fixed-points, and negations can be eliminated and reduced to a single formation of the least-fixed point of a positive first-order formula combined with the positive operations of first-order logic (disjunction, conjunction, universal and existential quantification).

6.5 Partial Fixed-Point Logic

The fundamental idea behind least fixed-point LFP is that recursive specifications involving positive first-order formulas can be given meaningful fixed-point semantics, because by the Knaster-Tarski Theorem 6.2, every positive first-order formula has a least fixed-point. Can more powerful logics be obtained by giving fixed-point semantics to specifications involving arbitrary (not just positive) first-order formulas? There are two main motivations behind this question that we now describe briefly.

Recall that on every class of finite structures, least fixed-point logic LFP is at least as expressive as first-order logic, but it is no more expressive than polynomial-time computability. In particular, on the class of all finite structures, LFP cannot express every polynomial-time computable query, even though it can express P-complete queries. As discussed at length in E. Grädel's Chapter in this volume, one of the outstanding open problems in finite model theory is whether or not there is *a logic that captures* P on the class of all finite structures. This has motivated the study of fixed-point logics that are at least as expressive as least fixed-point logic LFP, but are still withing the realm of polynomial-time computability on finite structures. Such a logic is *inflationary fixed-point logic* IFP, which, however, was shown by Gurevich and Shelah [GS86] to have the same expressive power as LFP on classes of finite structures (see E. Grädel's Chapter for the precise definitions of IFP and a presentation of some of its main properties on finite structures).

The second motivation for studying logics with more powerful fixed-point mechanisms has to do with the problem of finding logics that can express queries in higher computational complexty classes, beyond P and NP. The most prominent logic in this family is *partial fixed-point logic*, whose main features we will describe in what follows in the remainder of this section.

Let $\Phi : \mathcal{P}(A^k) \to \mathcal{P}(A^k)$ be an arbitrary (not necessarily monotone) k-ary operator on a finite set A. As seen earlier, the finite stages Φ^n, $n \geq 1$, of Φ are defined by the induction:

$$\left| \begin{array}{lcl} \Phi^1 & = & \Phi(\emptyset) \\ \Phi^{n+1} & = & \Phi(\Phi^n). \end{array} \right.$$

If Φ is not monotone, then the sequence Φ^n, $n \geq 1$, need not be an increasing one. Nonetheless, since A is a finite set and each Φ_n is a k-ary relation on A, there must exist two positive integers m and m' such that $m < m'$ and $\Phi^m = \Phi^{m'}$. Let m' be the smallest integer greater than m having

this property. If $m' = m+1$, then Φ^m is actually a fixed-point of Φ, and, thus, the sequence of stages of Φ converges to this fixed point. If, however, $m' > m+1$, then the sequence of stages of Φ cycles without ever reaching a fixed-point of Φ. This state of affairs motivates the concept of the *partial fixed-point* of an operator Φ.

Definition 6.34: Let $\Phi : \mathcal{P}(A^k) \to \mathcal{P}(A^k)$ be an arbitrary (not necessarily monotone) k-ary operator on a finite set A. The *partial fixed-point* $\mathbf{pfp}(\mathbf{\Phi})$ of Φ is a stage Φ^m such that $\Phi^m = \Phi^{m+1}$, if such a stage exists, or the empty relation $\emptyset$, otherwise.

If $\mathbf{A}$ is a a finite structure and Φ is the operator associated with some formula $\varphi(x_1, \ldots, x_k, S)$ on $\mathbf{A}$, then the *partial fixed-point* $\mathbf{pfp}(\varphi)$ of $\varphi(x_1, \ldots, x_k, S)$ is the partial fixed-point $\mathbf{pfp}(\mathbf{\Phi})$ of Φ.

Abiteboul and Vianu [AV91a] introduced *partial fixed-point logic* PFP on finite structures, which is the extension of first-order logic obtained by augmenting the syntax and the semantics with partial fixed-points of formulas.

Definition 6.35: Let σ be a vocabulary.

- The collection of PFP-formulas over σ is defined inductively by adding the following rule to the rules for the syntax of first-order logic:

  Assume that φ is a PFP-formula, S is a k-ary relation symbol in free(φ), $\mathbf{x} = (x_1, \ldots, x_k)$ is a k-tuple of first-order variables each of which is in free(φ), and $\mathbf{u} = (u_1, \ldots, u_k)$ is a k-tuple of first-order variables not occurring in φ. Then the expression $[\mathbf{pfp}\ S.\varphi](\mathbf{u})$ is a PFP-formula; moreover, free($[\mathbf{pfp}\ S.S](\mathbf{u})$) = free($\varphi$) $\setminus \{x_1 \ldots, x_k, S\}$.

- If $\mathbf{A}$ is a finite σ-structure, and $\mathbf{a}$ is a k-tuple from $\mathbf{A}$, then $\mathbf{A}, \mathbf{a} \models [\mathbf{pfp}\ S.\varphi](\mathbf{u})$ if $\mathbf{a} \in \mathbf{pfp}(\Phi)$, where Φ is the operator associated with φ on $\mathbf{A}$.

Clearly, if $\varphi(x_1, \ldots, x_k, S)$ is a formula that is positive in S, then the partial fixed-point $\mathbf{pfp}(\varphi)$ of φ is equal to its least fixed-point $\mathbf{lfp}(\varphi)$. It follows that on finite structures partial fixed-point logic PFP is at least as expressive as least fixed-point logic LFP. More precisely, if $\mathcal{C}$ is a class of finite structures, then

$$\text{LFP}(\mathcal{C}) \ \subseteq \ \text{PFP}(\mathcal{C}).$$

Let $\varphi(x_1, \ldots, x_k, S)$ be an arbitrary first-order formula over the vocabulary $\sigma \cup \{S\}$, where S is a k-ary relation symbol. It is easy to see that, on every finite structure $\mathbf{A}$, the partial fixed-point $\mathbf{pfp}(\varphi)$ can be evaluated in polynomial space. For this, one has to compute in succession the stages Φ^n of the operator Φ associated with $\varphi(x_1, \ldots, x_k, S)$, while at the same time maintaining a counter that stores in binary the number n of the current stage. At any given time in this computation, a polynomial amount of space is used to store the current stage Φ^n, to compute the next stage Φ^{n+1}, and to test whether $\Phi^{n+1} = \Phi^n$. If $\Phi^{n+1} = \Phi^n$, then the computation terminates and returns Φ^n as the value of the partial fixed-point $\mathbf{pfp}(\varphi)$ of $\varphi(x_1, \ldots, x_k, S)$ on $\mathbf{A}$. Otherwise, Φ^n is replaced by Φ^{n+1} and the counter is incremented by one. If at some point the value of the counter exceeds $2^{|A|^k}$ (which is the total number of k-ary relations on A), then the computation terminates and returns the empty relation $\emptyset$ as the value of the partial fixed-point $\mathbf{pfp}(\varphi)$ of $\varphi(x_1, \ldots, x_k, S)$ on $\mathbf{A}$. Thus, on every class $\mathcal{C}$ of finite structures, we have that

$$\text{LFP}(\mathcal{C}) \ \subseteq \ \text{PFP}(\mathcal{C}) \ \subseteq \ \text{PSPACE}.$$

The next example shows that PFP can actually express PSPACE-complete queries.

Example 6.36: Generalized Path Systems

Let σ be a vocabulary consisting of a unary relation symbol and a ternary relation symbol. As in the earlier Example 6.7, a σ-structure is of the form $\mathbf{S} = (F, A, R)$, where A is a subset of F and R is a ternary relation on F; moreover, such a structure can be interpreted as consisting of a set F of formulas, a set A of axioms, and a ternary rule of inference R. Let $\varphi(x, T)$ be the following existential first-order formula over the vocabulary $\sigma \cup \{T\}$:

$$A(x) \vee (\exists y)(\exists z)(T(y) \wedge \neg T(z) \wedge R(x, y, z)).$$

Intuitively, a fixed-point of this formula can be viewed as a recursive specification of a nonmonotonic proof system in which a formula x is a *theorem* of the system if it is an axiom in A or it can be derived from the rule of inference R using a theorem y of the proof system and a non-theorem z of the proof system.

Let $\mathbf{pfp}(\varphi)$ be the partial fixed-point of the formula $\varphi(x, T)$. Grohe [Gro95] showed that evaluating $\mathbf{pfp}(\varphi)$ on finite σ-structures is a PSPACE-complete problem. It follows that, unless P = PSPACE, the partial fixed-point $\mathbf{pfp}(\varphi)$ cannot be evaluated in polynomial time and, a fortiori, it cannot be expressed in LFP. ∎

The preceding remarks and Example 6.36 imply the following result concerning the data complexity of PFP.

Proposition 6.37: *The data complexity of* PFP *is* PSPACE*-complete.*

Let σ be a vocabulary containing at least one relation symbol of arity 2 or higher, and let $\mathcal{F}$ be the class of all finite σ-structures. Although PFP can express P-complete queries on $\mathcal{F}$, it cannot express every polynomial-time computable query on $\mathcal{F}$. Indeed, in the next section we will show that the expressive power of PFP on $\mathcal{F}$ has similar limitations as that of as LFP on $\mathcal{F}$, namely, PFP cannot express *counting* queries, such as Even Cardinality. Thus, the following proper containment holds on $\mathcal{F}$:

$$\text{PFP}(\mathcal{F}) \subset \text{PSPACE}.$$

The state of affairs, however, is different on classes of ordered finite structures.

Theorem 6.38: [Abiteboul-Vianu [AV91a], Vardi [Var82]]. *Let $\mathcal{C}$ be a class of ordered finite structures. The following are equivalent for a query Q on $\mathcal{C}$.*

- *Q is polynomial-space computable.*
- *Q is* PFP*-definable on $\mathcal{C}$.*

In other words, $\text{PSPACE}(\mathcal{C}) = \text{PFP}(\mathcal{C})$.

E. Grädel's Chapter in this volume contains a proof of the above theorem. Here, we discuss briefly the history of this result and explain the credits. Chandra and Harel [CH82] introduced and studied a logic called RQL, which is an extension of first order logic FO with recursion embodied in the form of WHILE looping. Vardi [Var82] proved that on classes of ordered finite structures, a query is polynomial-space computable if and only if it is RQL-definable. Later on, Abiteboul and Vianu [AV91a] introduced partial fixed-point logic PFP and showed that on classes of finite structures, RQL has the same expressive power as partial fixed-point logic PFP. From these results, it follows PSPACE = PFP on every class $\mathcal{C}$ of ordered finite structures.

In this section, we showed that on the class $\mathcal{F}$ of all finite σ-structures, LFP can express P-complete problems and PFP can express PSPACE-complete problems. At the same time, we asserted that these logics cannot express such basic counting properties as Even Cardinality on $\mathcal{F}$, but gave no proof of this fact. This will be done in the next section, where we will bring into the picture a family of infinitary logics with finitely many variables, will introduce new combinatorial games for analyzing their expressive power, and will apply the methodology of games to derive lower bounds for expressibility in fixed-point logics and in infinitary logics with finitely many variables.

7 Infinitary Logics with Finitely Many Variables

The syntax of the logics we have encountered thus far is finitary. Mathematical logicians, however, have also investigated in depth logics whose syntax has infinitary constructs. Such logics can be obtained by augmenting the syntax of first-order logic with disjunctions and conjunctions over infinite sets of formulas or with infinite strings of quantifiers or with both these types of constructs. Moreover, different families of infinitary logics can be obtained by imposing cardinality restrictions on the size of the infinitary constructs allowed (see [Kei71, Dic85]). The infinitary logic $\mathrm{L}_{\infty\omega}$ is the most powerful among all logics with infinitary connectives and with finite strings of quantifiers. In addition to the rules of first-order logic, the syntax of $\mathrm{L}_{\infty\omega}$ has the following two rules:

- If Φ is an arbitrary set of $\mathrm{L}_{\infty\omega}$-formulas, then the infinitary disjunction $\bigvee \Phi$ is also a $\mathrm{L}_{\infty\omega}$-formula.
- If Φ is an arbitrary set of $\mathrm{L}_{\infty\omega}$-formulas, then the infinitary conjunction $\bigwedge \Phi$ is also a $\mathrm{L}_{\infty\omega}$-formula.

The infinitary formulas $\bigvee \Phi$ and $\bigwedge \Phi$ have straightforward semantics. For instance, if Φ is a set of $\mathrm{L}_{\infty\omega}$-sentences and $\mathbf{A}$ is a structure, then $\mathbf{A} \models \bigvee \Phi$ if and only if there is at least one $\mathrm{L}_{\infty\omega}$-sentence φ in Φ such that $\mathbf{A} \models \varphi$.

Although $\mathrm{L}_{\infty\omega}$ can make interesting distinctions on infinite structures, it turns out that this logic is too powerful on classes of finite structures to be of any use. Specifically, it is easy to see that *every* Boolean query Q on the class $\mathcal{F}$ of all finite σ-structures is $\mathrm{L}_{\infty\omega}$-definable. Indeed, for every finite structure $\mathbf{A}$, let $\psi_{\mathbf{A}}$ be a first-order sentence that defines $\mathbf{A}$ up to isomorphism; such a sentence asserts that there are precisely as many elements as the cardinality of the universe A of $\mathbf{A}$, and states which tuples are in the relations of $\mathbf{A}$ and which are not. Since Boolean queries are closed under isomorphisms, Q is definable by the $\mathrm{L}_{\infty\omega}$-sentence $\bigvee_{\{\mathbf{A}:Q(\mathbf{A})=1\}} \psi_{\mathbf{A}}$. Note that Q is also definable by the $\mathrm{L}_{\infty\omega}$-sentence $\bigwedge_{\{\mathbf{A}:Q(\mathbf{A})=0\}} \neg\psi_{\mathbf{A}}$. Thus, every query on the class $\mathcal{F}$ of all finite σ-structures can be defined by both a countable disjunction of first-order formulas and a countable conjunctions of first-order formulas.

7.1 The Infinitary Logic $\mathrm{L}^{\omega}_{\infty\omega}$

In general, $\mathrm{L}_{\infty\omega}$-formulas may have an infinite number of distinct variables. Barwise [Bar77] introduced a family of fragments of $\mathrm{L}_{\infty\omega}$ in which there is a finite upper bound on the number of distinct variables in each formula.

Definition 7.1: Let σ be a vocabulary.

- For every positive integer k, we write $\mathrm{FO}^k_{\omega\omega}$ to denote the collection of all first-order formulas over σ with at most k distinct variables.

- For every positive integer k, the *k-variable infinitary logic* $\mathrm{L}^k_{\infty\omega}$ is the collection of all $\mathrm{L}_{\infty\omega}$-formulas over σ with at most k distinct variables.
- The *finite-variable infinitary logic* $\mathrm{L}^\omega_{\infty\omega}$ is the collection of all $\mathrm{L}_{\infty\omega}$-formulas over σ with finitely many variables, that is,

$$\mathrm{L}^\omega_{\infty\omega} \quad = \quad \bigcup_{k\geq 1} \mathrm{L}^k_{\infty\omega}.$$

Note that, although each $\mathrm{L}^k_{\infty\omega}$-formula has at most k distinct variables, there is no restriction on the number of occurrences of each variable in the formula. In particular, even $\mathrm{FO}^k_{\omega\omega}$-formulas may be of unbounded quantifier rank. In many cases, this makes it possible to define interesting properties by judiciously reusing the available variables, in spite of the limited supply of distinct variables. To illustrate this point, for every positive integer m, let θ_m be a first-order sentence asserting that there are at least m elements in the universe of the structure. It can be shown that on the class $\mathcal{G}$ of all finite graphs, θ_m is *not* equivalent to any first-order sentence with fewer than m variables. In contrast, it is easy to see that on the class $\mathcal{L}$ of all finite linear orders, θ_m is equivalent to a sentence of $\mathrm{L}^2_{\omega\omega}$. For instance, θ_4 is equivalent to the $\mathrm{L}^2_{\omega\omega}$-sentence

$$(\exists x)(\exists y)[y < x \wedge (\exists x)(x < y \wedge (\exists y)(y < x))].$$

It follows that $\mathrm{L}^2_{\infty\omega}$ can define arbitrary cardinalities on $\mathcal{L}$, since for every set S of integers we have that

$$n \in S \Longleftrightarrow \mathrm{L}_n \models \bigvee_{n\in S} (\theta_m \wedge \neg\theta_{m+1}).$$

In particular, the Even Cardinality query is $\mathrm{L}^2_{\infty\omega}$-definable on $\mathcal{L}$.

The original motivation behind the introduction of the finite-variable infinitary logic was to study inductive definability on fixed infinite structures. Indeed, Barwise [Bar77] used the infinitary logics $\mathrm{L}^k_{\infty\omega}$, $k \geq 1$, as a tool to solve an open problem concerning the closure ordinals of positive first-order formulas on fixed infinite structures. Since the 1980s, however, these logics have found many uses and applications in finite model theory, where they have become quite indispensable in the study of fixed-point logics. The main reason for this is that on classes of finite structures, $\mathrm{L}^\omega_{\infty\omega}$ subsumes the fixed-point logics LFP and PFP that we encountered earlier. Moreover, definability in the infinitary logics $\mathrm{L}^k_{\infty\omega}$, $k \geq 1$, can be characterized in terms of certain combinatorial games in a manner analogous to the characterization of first-order definability in terms of Ehrenfeucht-Fraïssé-games.

Before spelling out the connection between fixed-point logics and $\mathrm{L}^\omega_{\infty\omega}$ in more precise terms, we present a relevant example. Let $\varphi^n(x, y)$ be the first-order formula

$$(\exists z_1)\ldots(\exists z_{n-1})(E(x, z_1) \wedge \ldots \wedge E(z_{m-1}, y))$$

that defines the query "there is a path of length n from x to y", $n \geq 1$. At first sight, it appears that this query cannot be expressed with fewer than $n+1$ variables, since, in addition to the variables x and y, another $n-1$ variables seem to be needed in order to describe the intermediate nodes on a path of length n from x to y. It turns out, however, that, just three variables x, y and z suffice to express this query; the third variable z is repeatedly reused in such a way that it ranges over the intermediate points in a path from x to y. Specifically, it can be shown by induction on n that each

formula $\varphi^n(x,y)$ is equivalent to an $L^3_{\omega\omega}$-formula $\psi^n(x,y)$ whose variables are among x, y and z. First, $\psi^1(x,y)$ is equivalent to the atomic formula $E(x,y)$. Assume now that $\varphi^n(x,y)$ is equivalent to an $L^3_{\omega\omega}$-formula $\psi^n(x,y)$ whose variables are x, y and z. Then $\varphi^{n+1}(x,y)$ is equivalent to the $L^3_{\omega\omega}$-formula

$$(\exists z)[E(x,z) \wedge (\exists x)(z = x \wedge \psi^n(x,y))]$$

whose variables are x, y and z. Consequently, the CONNECTIVITY query is $L^3_{\infty\omega}$-definable by the sentence

$$(\forall x)(\forall y)(\bigvee_{n\geq 1} \psi^n(x,y)).$$

The preceding construction can be extended and applied to the stages of every first-order definable operator; this makes it possible to show that the stages of every first-order definable operator are definable by an $L^k_{\infty\omega}$-formula for some positive integer k that depends only on the formula defining the operator and not on the particular level of the stage. A detailed proof of the next result can be found in [KV92b, KV96]

Theorem 7.2: *Assume that σ is a vocabulary, S is a m-ary relation symbol not in σ, and $\varphi(x_1,\ldots,x_m,S)$ is a first-order formula over the vocabulary $\sigma \cup \{S\}$ such that the number of variables (free and bound) of $\varphi(x_1,\ldots,x_m,S)$ is equal to k.*

- *For every positive integer $n \geq 1$, there is a $FO^k_{\omega\omega}$-formula $\varphi^n(x_1,\ldots,x_m)$ that defines the n-th stage Φ^n of the operator Φ associated with $\varphi(x_1,\ldots,x_m,S)$ on every σ-structure.*
- *The partial fixed-point $\mathbf{pfp}(\varphi)$ of $\varphi(x_1,\ldots,x_m,S)$ is $L^k_{\infty\omega}$-definable on the class of all finite σ-structures.*

Consequently, if $\mathcal{C}$ is a class of finite σ-structures, then

$$\mathrm{LFP}(\mathcal{C}) \subseteq \mathrm{PFP}(\mathcal{C}) \subseteq L^\omega_{\infty\omega}(\mathcal{C}).$$

For arbitrary first-order formulas $\varphi(x_1,\ldots,x_m,S)$, only the finite stages of the associated operator were defined earlier. Recall, however, that if $\varphi(x_1,\ldots,x_m,S)$ is a positive first-order formula, then we actually defined the stages Φ^α of the associated operator Φ for an arbitrary ordinal α (Definition 6.1). It can be shown that each such stage Φ^α is definable by an $L^k_{\infty\omega}$-formula, where k is the number of variables (free and bound) in $\varphi(x_1,\ldots,x_m,S)$. It is not true, however, that every LFP-definable query on the class $\mathcal{S}$ of all σ-structures is $L^\omega_{\infty\omega}$-definable. For instance, the WELL FOUNDEDNESS query is LFP-definable on $\mathcal{S}$, but it is not $L_{\infty\omega}$-definable on $\mathcal{S}$ (see [Dic85]); consequently, this query is not $L^\omega_{\infty\omega}$-definable either. Intuitively, $L^\omega_{\infty\omega}$ can not subsume LFP on the class $\mathcal{S}$ of all σ-structures, because the closure ordinals of positive first-order formulas can be arbitrarily large and so the least fixed-point of a positive formula cannot be obtained by taking the disjunction over the formulas defining the stages of the formula (this would require taking a disjunction over a proper class, which is not allowed in the syntax of $L_{\infty\omega}$). It is true, however, that if $\mathcal{C}$ is a class of σ-structures of bounded cardinalities (that is, there is a cardinal number λ such that the universe of each structure in $\mathcal{C}$ has cardinality at most λ), then $\mathrm{LFP}(\mathcal{C}) \subseteq L^\omega_{\infty\omega}(\mathcal{C})$.

7.2 Pebble Games and $L^\omega_{\infty\omega}$-Definability

The finite-variable infinitary logic $L^\omega_{\infty\omega}$ can be used as a tool in studying fixed-point logics on finite structures. In particular, certain structural properties of $L^\omega_{\infty\omega}$ are inherited by the fixed-point logics

LFP and PFP. Moreover, lower bounds for definability in $L^{\omega}_{\infty\omega}$ yield immediately similar results for definability in LFP and PFP. The advantage of $L^{\omega}_{\infty\omega}$ over LFP and PFP is that, for each positive integer k, definability in $L^{k}_{\infty\omega}$ can be characterized in terms of combinatorial *k-pebble games* that we introduce next.

Definition 7.3: Let k be a positive integer, σ a vocabulary, and $\mathbf{A}$ and $\mathbf{B}$ two σ-steructures.

The *k-pebble game on* $\mathbf{A}$ *and* $\mathbf{B}$ is played between two players, called the *Spoiler* and the *Duplicator*, each of whom has k pebbles that are labeled $1, \ldots, k$. In each move, the Spoiler selects one of the two structures and either places a pebble that is not currently used on an element of the chosen structure or removes a pebble from an element of the chosen structure. The Duplicator responds by either placing the pebble with the same label on an element of the other structure or by removing the pebble with the same label from an element of the other structure.

Assume that at some point of time during the game, r pebbles have been placed on each structure, where $1 \leq r \leq k$, and let $(a_i, b_i) \in A \times B$, $1 \leq i \leq r$, be the pairs of elements of $\mathbf{A}$ and $\mathbf{B}$ such that the label of the pebble on a_i is the same as the label of the pebble on b_i. The *Spoiler wins the k-pebble game on* $\mathbf{A}$ *and* $\mathbf{B}$ at this point of time, if the mapping $a_i \mapsto b_i$, $1 \leq i \leq r$, is not an isomorphism between the substructures of $\mathbf{A}$ and $\mathbf{B}$ generated by $\{a_1, \ldots, a_r\}$ and $\{b_1, \ldots, b_r\}$, respectively.

The *Duplicator wins the k-pebble game on* $\mathbf{A}$ *and* $\mathbf{B}$ if the above never happens, which means that the Duplicator has a *winning strategy* that allows him to continue playing "forever" by maintaining a partial isomorphism at every point of time.

The above description of a winning strategy for the Duplicator in the k-pebble game is rather informal. The concept of a winning strategy can be made precise, however, in terms of families of partial isomorphisms with appropriate closure and extension properties. Recall that a partial isomorphism from a σ-structure $\mathbf{A}$ to a σ-structure $\mathbf{B}$ is an isomorphism from a substructure of $\mathbf{A}$ to a substructure of $\mathbf{B}$. In particular, every partial isomorphism from $\mathbf{A}$ to $\mathbf{B}$ must map each constant $c^{\mathbf{A}}_j$ of $\mathbf{A}$ to the constant $c^{\mathbf{B}}_j$, $1 \leq j \leq s$. Thus, when viewed as a set of ordered pairs, each partial isomorphism from $\mathbf{A}$ to $\mathbf{B}$ must contain all pairs $(c^{\mathbf{A}}_j, c^{\mathbf{B}}_j)$, $1 \leq j \leq s$.

Definition 7.4: A *winning strategy for the Duplicator in the k-pebble game on* $\mathbf{A}$ *and* $\mathbf{B}$ is a nonempty family $\mathcal{I}$ of partial isomorphisms from $\mathbf{A}$ to $\mathbf{B}$ with the following properties.

1. If $f \in \mathcal{I}$, then $|f - \{(c^{\mathbf{A}}_1, c^{\mathbf{B}}_1) \ldots, (c^{\mathbf{A}}_1, c^{\mathbf{B}}_s)\}| \leq k$.
2. $\mathcal{I}$ is *closed under subfunctions*:
   If $g \in \mathcal{I}$ and f is a function such that $\{(c_1,^{\mathbf{A}}, c^{\mathbf{B}}_1), \ldots, (c^{\mathbf{A}}_s, c^{\mathbf{B}}_s)\} \subseteq f \subseteq g$, then $f \in \mathcal{I}$.
3. $\mathcal{I}$ has the *forth property up to k*:
   If $f \in \mathcal{I}$ and $|f - \{(c^{\mathbf{A}}_1, c^{\mathbf{B}}_1) \ldots, (c^{\mathbf{A}}_1, c^{\mathbf{B}}_s)\}| < k$, then for every $a \in A$, there is $g \in \mathcal{I}$ so that $f \subseteq g$ and $a \in \mathrm{dom}(g)$.
4. $\mathcal{I}$ has the *back property up to k*:
   If $f \in \mathcal{I}$ and $|f - \{(c^{\mathbf{A}}_1, c^{\mathbf{B}}_1) \ldots, (c^{\mathbf{A}}_1, c^{\mathbf{B}}_s)\}| < k$, then for every $b \in B$, there is $g \in \mathcal{I}$ so that $f \subseteq g$ and $b \in \mathrm{rng}(g)$.

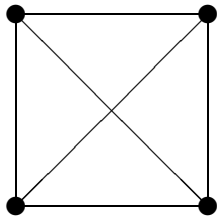
Intuitively, the second condition provides the Duplicator with a "good" move when the Spoiler removes a pebble from an element of $\mathbf{A}$ or $\mathbf{B}$, while the last two conditions provide the Duplicator with "good" moves when the Spoiler places a pebble on an element of $\mathbf{A}$ or on an element of $\mathbf{B}$.

Several properties of the k-pebble game follow easily from the definitions. For instance, if $k' \geq k$ and the Spoiler wins the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$, then the Spolier also wins the k'-pebble game on $\mathbf{A}$ and $\mathbf{B}$. Moreover, for every $k \geq 1$, the relation "the Duplicator wins the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$ is an equivalence relation on the class $\mathcal{S}$ of all σ-structures.

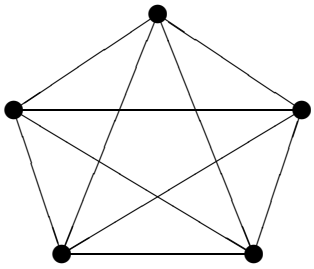
The following examples illustrate k-pebble games on concrete finite structures.

Example 7.5: For every $m \geq 1$, let $\mathbf{K}_m$ be the m-clique, that is, the complete graph with m nodes. It is quite clear that for every $k \geq 1$:

- The Duplicator wins the k-pebble game on $\mathbf{K}_k$ and $\mathbf{K}_{k+1}$.
- The Spoiler wins the $(k+1)$-pebble game on $\mathbf{K}_k$ and $\mathbf{K}_{k+1}$.



$\mathbf{K}_4$



$\mathbf{K}_5$

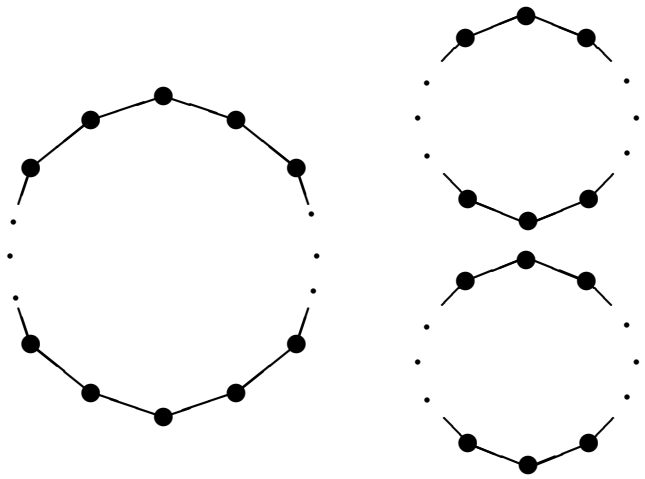
Note that the same state of affairs holds for the Ehrenfeucht-Fraïssé-game on cliques: the Duplicator wins the k-move Ehrenfeucht-Fraïssé-game on $\mathbf{K}_k$ and $\mathbf{K}_{k+1}$, but the Spoiler wins the $(k+1)$-move Ehrenfeucht-Fraïssé-game on $\mathbf{K}_k$ and $\mathbf{K}_{k+1}$. ∎

Example 7.6: For every $m \geq 1$, let $\mathbf{L}_m$ be the linear order with m elements.

It is easy to see that for all positive integers m and n with $m < n$, the Spoiler wins the 2-pebble game on $\mathbf{L}_m$ and $\mathbf{L}_n$. Indeed, in the first two moves, the Spoiler places his two pebbles on the two smallest elements of $\mathbf{L}_n$; it is then to the best interest of the Duplicator to place his two pebbles in the two smallest elements of L_m. In his next two movers, the Spoiler moves the pebble from the smallest element of $\mathbf{L}_n$ and places it on the third smallest element of $\mathbf{L}_n$; the Duplicator has to follow suit with similar moves on $\mathbf{L}_m$. By continuing playing this way, the Spoiler forces the placement of pebbles with the same lable in progressively bigger elements of the two linear orders. Since $m < n$, eventually the Duplicator "runs out of elements" in $\mathbf{L}_m$ and cannot duplicate the move of the Spoiler.

In view of Theorem 3.20, this example shows a dramatic difference between the pebble games and the Ehrenfeucht-Fraïssé-games, since for every r and for all sufficiently large m and n, the Duplicator wins the r-move Ehrenfeucht-Fraïssé-game on $\mathbf{L}_m$ and $\mathbf{L}_n$. ∎

Example 7.7: For every $m \geq 3$, let $\mathbf{A}_m$ be a directed cycle with $2m$ nodes and $\mathbf{B}_m$ be the union of two disjoint directed cycles each with m nodes.



$\mathbf{A}_m$ $\mathbf{B}_m$

It is easy to see that for every $m \geq 3$, the Spoiler wins the 3-pebble game on $\mathbf{A}_m$ and $\mathbf{B}_m$. Indeed, in the first two moves, the Spoiler places his first pebble on a node in the top cycle of $\mathbf{B}_m$ and his second pebble on a node in the bottom cycle of $\mathbf{B}_m$; thus, the Duplicator has to respond by placing his first two pebbles on elements of $\mathbf{A}_m$ (presumably, as far apart as possible). From this point on, the Spoiler keeps his first pebble fixed on the top cycle, but uses his second and third pebbles to force a walk along edges of the bottom cycle, the same way as the Spoiler moved from smaller to bigger elements in the preceding Example 7.6. Eventually, the three pebbles of the Duplicator are lined up along adjacent nodes in $\mathbf{A}_m$, but this doens not hold for the pebbles of the Spoiler in $\mathbf{B}_m$.

This example should be contrasted with the fact that, as implied by the proof of Proposition 3.28, for every $r \geq 1$ and for all sufficiently large values of m, the Duplicator wins the r-move Ehrenfeucht-Fraïssé-game on $\mathbf{A}_m$ and $\mathbf{B}_m$. ∎

We are now ready to present the connection between k-pebble games and definability in the k-variable infinitary logics $\mathrm{L}^k_{\infty\omega}$, $k \geq 1$.

Definition 7.8: Let k be a positive integer, and let $\mathbf{A}$ and $\mathbf{B}$ be two σ-structures.

- $\mathbf{A}$ is $\mathrm{L}^k_{\infty\omega}$-*equivalent* to $\mathbf{B}$, denoted by $\mathbf{A} \equiv^k_{\infty\omega} \mathbf{B}$, if $\mathbf{A}$ and $\mathbf{B}$ satisfy the same $\mathrm{L}^k_{\infty\omega}$-sentences.
- We write $\mathbf{A} \equiv^k_{\omega\omega} \mathbf{B}$ to denote that $\mathbf{A}$ and $\mathbf{B}$ satisfy the same $\mathrm{FO}^k_{\omega\omega}$-sentences.
- Let $a_1, \ldots, a_r$ be a sequence of elements from A and let $b_1, \ldots, b_r$ be a sequenec of elements from B, for some $r \leq k$.

  $(\mathbf{A}, a_1, \ldots, a_r)$ is $\mathrm{L}^k_{\infty\omega}$-*equivalent* to $(\mathbf{B}, b_1, \ldots, b_r)$, denoted $(\mathbf{A}, a_1, \ldots, a_r) \equiv^k_{\infty\omega} (\mathbf{B}, b_1, \ldots, b_r)$, if for every $\mathrm{L}^k_{\infty\omega}$-formula $\varphi(v_1, \ldots, v_r)$ with free variables among $v_1, \ldots, v_r$, we have that

$$\mathbf{A} \models \varphi(v_1/a_1, \ldots, v_r/a_r) \iff \mathbf{B} \models \varphi(v_1/a_1, \ldots, v_r/b_r).$$

Clearly, $\equiv^k_{\infty\omega}$ is an equivalence relation on the class $\mathcal{S}$ of all σ-structures, which we call $\mathrm{L}^k_{\infty\omega}$-*equivalence*. The next result asserts that $\mathrm{L}^k_{\infty\omega}$-equivalence coincides with the equivalence relation that arises from the k-pebble game. Here, we only outline the main ideas of the proof; complete details can be found in [KV92b]

Theorem 7.9: [Bar77, Imm82] *Let k be a positive integer, and let $\mathbf{A}$ and $\mathbf{B}$ be two σ-structures. Then the following statements are equivalent:*

- $\mathbf{A} \equiv^k_{\infty\omega} \mathbf{B}$.
- *The Duplicator wins the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$.*

Moreover, if $\mathbf{A}$ and $\mathbf{B}$ are finite, then the above statements are also equivalent to

- $\mathbf{A} \equiv^k_{\omega\omega} \mathbf{B}$.

Proof: *(Outline)* Assume first that $\mathbf{A}$ and $\mathbf{B}$ are two σ-structures such that $\mathbf{A} \equiv^k_{\infty\omega} \mathbf{B}$. We have to show that there is a family $\mathcal{I}$ of partial isomorphisms on $\mathbf{A}$ and $\mathbf{B}$ that provides a winning strategy for Player II in the k-pebble game, as described in Definition 7.4.

We take $\mathcal{I}$ to be the family of all partial isomorphisms f between $\mathbf{A}$ and $\mathbf{B}$ such that the following hold:

- $|f - \{(c_1^{\mathbf{A}}, c_1^{\mathbf{B}}), \ldots, (c_1^{\mathbf{A}}, c_s^{\mathbf{B}})\}| \leq k$.
- If $a_1, \ldots, a_r$ are all elements in the domain of f other than the elements $c_1^{\mathbf{A}}, \ldots, c_s^{\mathbf{A}}$ (that interpret the constant symbols) and $b_1 = f(a_1), \ldots, b_r = f(a_r)$ are their images under f, then $(\mathbf{A}, a_1, \ldots, a_r) \equiv^k_{\infty\omega} (\mathbf{B}, b_1, \ldots, b_r)$.

To show that $\mathcal{I}$ is a winnning strategy for the Duplicator, first note that $\mathcal{I}$ is non-empty, because $\mathbf{A} \equiv^k_{\infty\omega} \mathbf{B}$ and, thus, the function f with $f(c_j^{\mathbf{A}}) = c_j^{\mathbf{B}}$, $1 \leq j \leq s$, is a member of $\mathcal{I}$. Moreover, $\mathcal{I}$ is clearly closed under subfunctions. To show that $\mathcal{I}$ has the forth property up to k, it suffices to show that for all $r < k$, if we have two sequences of distinct elements $a_1, \ldots, a_r$ in A and $b_1, \ldots, b_r$ in B such that

$$(\mathbf{A}, a_1, \ldots, a_r) \equiv^k_{\infty\omega} (\mathbf{B}, b_1, \ldots, b_r),$$

then for every element a in A that is different from $a_1, \ldots, a_r$ there is an element b in B that is different from $b_1, \ldots, b_r$ and is such that

$$(\mathbf{A}, a_1, \ldots, a_r, a) \equiv^k_{\infty\omega} (\mathbf{B}, b_1, \ldots, b_r, b).$$

Assume that no such $b \in B$ exists for a certain $a \in A$. Then for every $b \in B$ that is different from $b_1, \ldots, b_r$ there is an $\mathrm{L}^k_{\infty\omega}$-formula $\psi_b(v_1, \ldots, v_r, v)$ such that

$$(\mathbf{A}, a_1, \ldots, a_r, a) \models \psi_b(v_1, \ldots, v_r, v)$$

and

$$(\mathbf{B}, b_1, \ldots, b_r, b) \not\models \psi_b(v_1, \ldots, v_r, v).$$

Hence,

$$(\mathbf{A}, a_1, \ldots, a_r) \models (\exists v) \left((v_1 \neq v) \wedge \ldots \wedge (v_r \neq v) \wedge \bigwedge_{b \in B} \psi_b(v_1, \ldots, v_r, v) \right),$$

and, at the same time,

$$(\mathbf{B}, b_1, \ldots, b_r) \not\models (\exists v) \left((v_1 \neq v) \wedge \ldots \wedge (v_r \neq v) \wedge \bigwedge_{b \in B} \psi_b(v_1, \ldots, v_m, v) \right).$$

But this is a contradiction, since

$$(\exists v)\left((v_1 \neq v) \wedge \ldots \wedge (v_m \neq v) \wedge \bigwedge_{b \in B} \psi_b(v_1, \ldots, v_m, v)\right)$$

is an $\mathrm{L}^k_{\infty\omega}$-formula and $(\mathbf{A}, a_1, \ldots, a_r) \equiv^k_{\infty\omega} (\mathbf{B}, b_1, \ldots, b_r)$. The forth property up to k is established in an analogous manner, using an infinitary conjunction over elements of A. Note that if $\mathbf{A}$ and $\mathbf{B}$ are finite σ-structures, then these conjunctions are actually finitary. Using this observation, we can mimick the preceding argument with $\equiv^k_{\omega\omega}$ in place of $\equiv^k_{\infty\omega}$ in the definition of the winning strategy $\mathcal{I}$. It follows that if $\mathbf{A}$ and $\mathbf{B}$ are finite σ-structures satisfying the same $\mathrm{FO}^k_{\omega\omega}$-sentences, then the Duplicator wins the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$.

Conversely, let $\mathcal{I}$ be a winning strategy for the Duplicator in the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$. We have to show that $\mathbf{A}$ and $\mathbf{B}$ satisfy the same $\mathrm{L}^k_{\infty\omega}$-sentences. This is a consequence of the following stronger statement, which can be proved by induction on the construction of $\mathrm{L}^k_{\infty\omega}$-formulas using the closure and extension properties of $\mathcal{I}$:

If $\psi(v_1, \ldots, v_r)$ is an $\mathrm{L}^k_{\infty\omega}$-formula whose variables are among $v_1, \ldots, v_k$ and whose free variables are among $v_1, \ldots, v_r$, then for all $f \in \mathcal{I}$ and for all (not necessarily distinct) elements $a_1, \ldots, a_r$ from the domain of f, we have

$$\mathbf{A} \models \psi(v_1/a_1, \ldots, v_r/a_r) \Longleftrightarrow \mathbf{B} \models \psi(v_1/f(a_1), \ldots, v_m/f(a_r)).$$

∎

As a consequence of Theorem 7.9, we obtain a characterization of $\mathrm{L}^\omega_{\infty\omega}$-definability on classes of finite structures.

Corollary 7.10: *Let σ be a vocabulary, $\mathcal{C}$ a class of finite σ-structures, and Q a Boolean query on $\mathcal{C}$. Then the following statements are equivalent:*

1. *Q is $\mathrm{L}^\omega_{\infty\omega}$-definable on $\mathcal{C}$.*
2. *There is a positive integer k such that for every structure $\mathbf{A} \in \mathcal{C}$ and every structure $\mathbf{B} \in \mathcal{C}$, if $Q(\mathbf{A}) = 1$ and the Duplicator wins the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$, then $Q(\mathbf{B}) = 1$.*

Proof: If Q is $\mathrm{L}^\omega_{\infty\omega}$-definable on $\mathcal{C}$, then there is a positive integer k such that Q is definable on $\mathcal{C}$ by some $\mathrm{L}^k_{\infty\omega}$-sentence θ. Theorem 7.9 implies that if $\mathbf{A}$ and $\mathbf{B}$ are structures in $\mathcal{C}$ such that $Q(\mathbf{A}) = 1$ and the Duplicator wins the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$, then $\mathbf{B} \models \theta$, hence $Q(\mathbf{A}) = 1$. Note that the assumption that $\mathcal{C}$ consists of finite structures was not used in this direction.

For the other direction, assume that k is a positive integer with the property that if $\mathbf{A}$ and $\mathbf{B}$ are structures in $\mathcal{C}$ such that $Q(\mathbf{A}) = 1$ and the Duplicator wins the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$, then $Q(\mathbf{B}) = 1$. For every structure $\mathbf{A} \in \mathcal{C}$, let $\Psi_{\mathbf{A}}$ be the set of all $\mathrm{FO}^k_{\omega\omega}$-sentences ψ such that $\mathbf{A} \models \psi$. Note that $\Psi_{\mathbf{A}}$ is actually a countable set because there are countably many first-order formulas; consequently, $\bigwedge \Psi_{\mathbf{A}}$ is an $\mathrm{L}^k_{\infty\omega}$-sentence. Let $\mathbf{A}_1, \ldots, \mathbf{A}_n, \ldots$ be a list of representatives of all isomorphism types of structures in $\mathcal{C}$. Such a list is countable, since there are countably many non-isomorphic finite structures. Using Theorem 7.9, it is easy to see that the $\mathrm{L}^k_{\infty\omega}$-sentence $\bigvee\{\bigwedge \Psi_{\mathbf{A}_n} : n \geq 1\}$ defines the query Q on $\mathcal{C}$. ∎

Method 7.11: The Method of k-Pebble Games for $\mathrm{L}^\omega_{\infty\omega}$
Let σ be a vocaculary, $\mathcal{C}$ a class of finite σ-structures, and Q a Boolean query on $\mathcal{C}$.

Soundness: To show that Q is *not* $\mathrm{L}^{\omega}_{\infty\omega}$-definable on $\mathcal{C}$, it suffices to show that for every positive integer k there are structures $\mathbf{A}_k$ and $\mathbf{B}_k$ in $\mathcal{C}$ such that

- $Q(\mathbf{A}_k) = 1$ and $Q(\mathbf{B}_k) = 0$.
- The Duplicator wins the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$.

Completeness: This method is also complete, that is, if Q is *not* $\mathrm{L}^{\omega}_{\infty\omega}$-definable on $\mathcal{C}$, then for every positive integer k such structures $\mathbf{A}_k$ and $\mathbf{B}_k$ exist. ∎

We note that the above method is sound for arbitrary classes of σ-structures, not just classes of finite σ-structures. Moreover, it can be shown that it is complete for classes of σ-structures of bounded cardinalities. We now present some applications of this method.

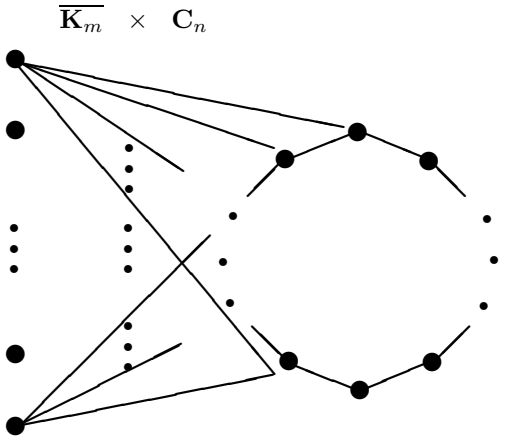
Proposition 7.12: *Let $\mathcal{G}$ be the class of all finite graphs.*

- *The* Even Cardinality *query is not $\mathrm{L}^{\omega}_{\infty\omega}$-definable on $\mathcal{G}$. Consequently, the* Even Cardinality *query is neither* LFP*-definable nor* PFP*-definable on $\mathcal{G}$.*
- *For every $k \geq 1$, the query "does the graph contain a $(k+1)$-clique?" is not $\mathrm{L}^{k}_{\infty\omega}$-definable on $\mathcal{G}$.*

Proof: This is an immediate consequence of Example 7.5, Theorem 7.10, and Theorem 7.2. ∎

Proposition 7.13: (de Rougemont [dR87]) *The query* Hamiltonian Path *is not $\mathrm{L}^{\omega}_{\infty\omega}$-definable on the class $\mathcal{G}$ of all finite graphs. Consequently, the query* Hamiltonian Path *is neither* LFP*-definable nor* PFP*-definable on $\mathcal{G}$.*

Proof: For every $m \geq 1$ and every $n \geq 1$, let $\overline{\mathbf{K}}_m \times \mathbf{C}_n$ be the product graph of the totally disconnected m-node graph $\mathbf{K}_m$ with the n-node cycle $\mathbf{C}_n$, as depicted below.



It is easy to see that $\overline{\mathbf{K}}_m \times \mathbf{C}_n$ has a Hamiltonian Path if and only if $m \leq n$. Indeed, this holds because, in order to visit two nodes of $\mathbf{K}_m$ by traveling along edges of $\overline{\mathbf{K}}_m \times \mathbf{C}_n$, one has to visit a node of $\mathbf{C}_n$. Moreover, it is quite clear that for every $k \geq 1$, the Duplicator wins the k-pebble

game on $\overline{\mathbf{K}}_k \times \mathbf{C}_k$ and $\overline{\mathbf{K}}_{k+1} \times \mathbf{C}_k$. Since $\overline{\mathbf{K}}_k \times \mathbf{C}_k$ has a HAMILTONIAN PATH, but $\overline{\mathbf{K}}_{k+1} \times \mathbf{C}_k$, the conclusions follow immediately from Theorem 7.10, and Theorem 7.2. ∎

As an exercise, we invite the reader to apply Method 7.11 and show that the PERFECT MATCHING query is not $\mathrm{L}^{\omega}_{\infty\omega}$-definable on $\mathcal{G}$. Note that, using the same method, Dawar [Daw98] showed that 3-COLORABILITY is not $\mathrm{L}^{\omega}_{\infty\omega}$-definable on $\mathcal{G}$. This is a technically difficult result that requires the construction of complicated graphs $\mathbf{A}_k$ and $\mathbf{B}_k$, $k \geq 1$, such that $\mathbf{A}_k$ is 3-colorable, $\mathbf{B}$ is not 3-colorable, and the Duplicator wins the k-pebble game on $\mathbf{A}_k$ and $\mathbf{B}_k$. In contrast, 2-COLORABILITY is $\mathrm{L}^{\omega}_{\infty\omega}$-definable on $\mathcal{G}$; in fact, it is $\mathrm{L}^{4}_{\infty\omega}$-definable, since NON-2-COLORABILITY is definable by a Datalog program with at most four variables in each rule, as shown in Example 6.20.

It should be pointed out that, although Method 7.11 can be used to establish limitations on the expressive power of LFP and PFP on the class $\mathcal{G}$ of all finite graphs, this method cannot be used to establish such results on the class $\mathcal{O}$ of all finite ordered graphs. The reason for this is that *every* query on $\mathcal{O}$ is $\mathrm{L}^{2}_{\infty\omega}$-definable, since the isomorphism type of every ordered finite structure is definable by an $\mathrm{L}^{2}_{\infty\omega}$-sentence (this is an extension of the fact that $\mathrm{L}^{2}_{\omega\omega}$ can express every fixed finite cardinality on linear orders). In particular, HAMILTONIAN PATH and 3-COLORABILITY are $\mathrm{L}^{2}\infty\omega$-expressible on $\mathcal{O}$. Consequently, Method 7.11 cannot be used to establish limitations on the expressive power of LFP and PFP on the class $\mathcal{O}$ of all ordered finite graphs; this is not surprising, since, as stated in Theorem 6.11 and Theorem 6.38, LFP capturea PTIME and PFP captures PSPACE on $\mathcal{O}$.

Up to this point, k-pebble games have been used to establish mainly *negative* results, that is, lower bounds for definability in $\mathrm{L}^{\omega}_{\infty\omega}$ and, a fortiori, lower bounds for definability in LFP and in PFP. These games, however, can also be used to establish *positive* results in the form of structural properties of $\mathrm{L}^{\omega}_{\infty\omega}$ that in many cases are inherited by LFP and PFP. Moreover, k-pebble games can be used to unveil certain deeper connections between LFP and $\mathrm{L}^{\omega}_{\infty\omega}$. As will be seen in what follows in the remainder of this section, all these results involve an in-depth study of the family of the equivalence relations $\equiv^{k}_{\infty\omega}$, $k \geq 1$, using k-pebble games.

7.3 0-1 Laws for $\mathrm{L}^{\omega}_{\infty\omega}$

A major direction of research in finite model theory has focused on the study of the asymptotic probabilities of queries on classes of finite structures. This is the topic of the Chapter by Joel Spencer in this volume. Here, we present a brief overview of 0-1 laws for the infinitary logic $\mathrm{L}^{\omega}_{\infty\omega}$.

Definition 7.14: Let σ be a vocabulary, $\mathcal{C}$ a class of finite σ-structures, and Q a Boolean query on $\mathcal{C}$.

- For every $n \geq 1$, we write $\mathcal{C}_n$ to denote the subclass of $\mathcal{C}$ consisting of all structures $\mathbf{A}$ in $\mathcal{C}$ such that the universe of $\mathbf{A}$ has cardinality n.
- For every $n \geq 1$, let μ_n be a probability measure on $\mathcal{C}_n$.
  - We write $\mu_n(Q)$ to denote the probability of the query Q on $\mathcal{C}_n$ with respect to the measure μ_n, $n \geq 1$.
  - The *asymptotic probability* $\mu(Q)$ *of the query* Q *with respect to the family of measures* μ_n, $n \geq 1$ is defined as

$$\mu(Q) \quad = \quad \lim_{n \to \infty} \mu_n(Q),$$

provided the limit exists.

Among all measures on classes of finite structures, the *uniform measure* is the most well-studied one. More precisely, if $\mathcal{C}$ is a class of finite σ-structures and Q is a Boolean query on $\mathcal{C}$, then the value $\mu_n(Q)$ of the uniform measure is equal to the fraction of structures in $\mathcal{C}_n$ that satisfy the query Q, $n \geq 1$. Combinatorialists have studied in depth the asymptotic probabilities of queries on finite graphs with respect to the uniform measure. For instance, it is well known that $\mu(\text{4-REGULAR}) = 0$, $\mu(\text{2-COLORABILITY}) = 0$, and $\mu(\text{HAMILTONIAN PATH}) = 1$. Note, however, that $\mu(\text{EVEN CARDINALITY})$ does *not* exist, since $\mu_{2n}(\text{EVEN CARDINALITY}) = 1$ and $\mu_{2n+1}(\text{EVEN CARDINALITY}) = 0$.

In the late 1960s and early 1970s, researchers raised the question of whether there is a connection between the definability of a query Q in some logic and its asymptotic probability with respect to a given measure. The next definition captures a case in which such a connection exists and it is tight.

Definition 7.15: Let L be a logic, σ a vocabulary consisting of relation symbols only, $\mathcal{C}$ a class of finite σ-structures, and μ_n, $n \geq 1$, a family of measures on $\mathcal{C}_n$.

We say that L has a *0-1 law on $\mathcal{C}$ with respect to μ_n, $n \geq 1$*, if for every $\mathcal{L}$-definable query Q on $\mathcal{C}$, we have that $\mu(Q) = 0$ or $\mu(Q) = 1$.

Note that the presence of constant symbols causes the failure of the 0-1 law for first-order logic with respect to the uniform measure. Indeed, if σ is a vocabulary containing a constant symbol c and a unary relation symbol P, then it is quite easy to verity that $\mu(P(c)) = 1/2$. This explains why in Definition 7.15 it was assumed that the vocabulary consists of relation symbols only.

Over the years, there has been an extensive investigation of 0-1 laws for various logics with respect to the uniform measure on classes of finite σ-structures and, in particular, on the class $\mathcal{F}$ of all finite σ-structures. This investigation started with the independent discovery by Glebskii et al. [GKLT69] and Fagin [Fag76] that first-order logic FO has a 0-1 law with the respect to the uniform measure on the class $\mathcal{F}$ of all finite σ-structures. After this, Blass, Gurevich and Kozen [BGK85] showed that least fixed-point logic LFP has a 0-1 law with respect to the uniform measure on $\mathcal{F}$, while Kolaitis and Vardi [KV87] showed that partial fixed-point logic PFP has a 0-1 law with respect to the uniform measure on $\mathcal{F}$. These 0-1 laws for progressively more expressive logics turned out to be special cases of the 0-1 law for the infinitary logic $\mathrm{L}^{\omega}_{\infty\omega}$ with respect to the uniform measure on $\mathcal{F}$, a result established by Kolaitis and Vardi [KV92b].

Theorem 7.16: *Let σ be a vocabulary consisting of relation symbols only. Then the finite-variable infinitary logic $\mathrm{L}^{\omega}_{\infty\omega}$ has a 0-1 law with respect to the uniform measure on the class $\mathcal{F}$ of all finite σ-structures.*

Proof: *(Hint)* For every $k \geq 1$, let θ_k be the conjunction of all *extension axioms* for σ with at most k variables, that is, the conjunction of all $\mathrm{FO}^k_{\omega\omega}$-sentences that assert that every substructure with fewer than k elements has an extension to a substructure with k elements. Fagin [Fag76] showed that $\mu(\theta_k) = 1$, where μ_n is the uniform measure on $\mathcal{F}_n$, $n \geq 1$. Let $\mathbf{A}_k$ be a model of θ_k, and let $[\mathbf{A}_k]_{\equiv^k_{\infty\omega}} = \{\mathbf{B} \in \mathcal{F} : \mathbf{A}_k \equiv^k_{\infty\omega} \mathbf{B}\}$ be the $\equiv^k_{\infty\omega}$-equivalence class of $\mathbf{A}_k$. Using the characterization of $\equiv^k_{\infty\omega}$ via k-pebble games in Theorem 7.9, it can be shown that $[\mathbf{A}_k]_{\equiv^k_{\infty\omega}} = \{\mathbf{B} \in \mathcal{F} : \mathbf{B} \models \theta_k\}$. Consequently, $\mu([\mathbf{A}_k]_{\equiv^k_{\infty\omega}}) = 1$, which easily implies that the 0-1 law holds for the k-variable infinitary logic $\mathrm{L}^k_{\infty\omega}$. ∎

Since the asymptotic probability of the EVEN CARDINALITY query does not exist, the preceding Theorem 7.16 gives another proof that EVEN CARDINALITY is not $\mathrm{L}^{\omega}_{\infty\omega}$-definable on the class $\mathcal{F}$ of all finite σ-structures.

The next result characterizes when a 0-1 law holds for the k-variable infinitary logics $\mathrm{L}^k_{\infty\omega}$, $k \geq 1$, on a class of finite structures with respect to an arbitrary measure.

Theorem 7.17: *Let σ be a vocabulary consisting of relation symbols only, $\mathcal{C}$ a class of finite σ-structures, and μ_n a measure on $\mathcal{C}_n$, $n \geq 1$. Then, for every positive integer k, the following two statements are equivalent:*

1. *The k-variable infinitary logic $\mathrm{L}^k_{\infty\omega}$ has a 0-1 law with respect to μ_n, $n \geq 1$, on $\mathcal{C}$.*
2. *There is an equivalence class D of $\equiv^k_{\infty\omega}$ on $\mathcal{C}$ such that $\mu(D) = 1$.*

In effect, Theorem 7.17 reveals that $\mathrm{L}^k_{\infty\omega}$ has a 0-1 law if and only if there is a "giant" $\mathrm{L}^k_{\infty\omega}$-equivalance class; all other $\mathrm{L}^k_{\infty\omega}$-equivalence classes must have asymptotic probability equal to 0. Moreover, the existence of a 0-1 law for $\mathrm{L}^k_{\infty\omega}$ can be established using k-pebble games.

As described in Joel Spencer's Chapter in this volume, Shelah and Spencer [SS88] investigated 0-1 laws for first-order logic FO on the class $\mathcal{G}$ of all finite graphs under non-uniform measures on $\mathcal{G}_n$ of the form $p(n) = n^{-\alpha}$, where α is a fixed real number. Their main finding is that first-order logic FO has a 0-1 law on $\mathcal{G}$ with respect to the measures $p(n) = n^{-\alpha}$ if and only if α is an irrational number. It follows that if α is a rational number, then the 0-1 law fails for the finite-variable infinitary logic $\mathrm{L}^\omega_{\infty\omega}$ on $\mathcal{G}$ with respect to the measures $p(n) = n^{-\alpha}$. Moreover, McArthur [McA95] showed that the 0-1 law fails for $\mathrm{L}^\omega_{\infty\omega}$ on $\mathcal{G}$ with respect to the measures $p(n) = n^{-\alpha}$ also when α is an irrational number. Thus, the 0-1 law fails for $\mathrm{L}^\omega_{\infty\omega}$ on $\mathcal{G}$ with respect to every measure of the form $p(n) = n^{-\alpha}$.

7.4 Definability and Complexity of $\mathrm{L}^k_{\infty\omega}$-Equivalence

If L is a logic and σ is a vocabulary, then two σ-structures $\mathbf{A}$ and $\mathbf{B}$ are *L-equivalent* if they satisfy the same L-sentences. The concept of L-equivalence gives rise to the following decision problem: given two finite σ-structures $\mathbf{A}$ and $\mathbf{B}$, are they L-equivalent? Strictly speaking, this decision problem is not a query on finite structures, since, according to Definition 2.1, queries take single structures as inputs, not pairs of structures. It is easy, however, to view this decision problem as a query on an expanded vocabulary $\sigma_1 + \sigma_2$ that consists of two disjoint copies of the relation and constant symbols in the vocabulary σ together with two unary predicates D_1 and D_2. Using the vocabulary $\sigma_1 + \sigma_2$, a pair (A, B) of two σ-structures $\mathbf{A}$ and $\mathbf{B}$ is identified with a single $\sigma_1 + \sigma_2$-structure $\mathbf{A} + \mathbf{B}$ defined as follows: the universe of $\mathbf{A} + \mathbf{B}$ is the union $A \cup B$ of the universes of $\mathbf{A}$ and $\mathbf{B}$, the relation symbol D_1 is interpreted by the universe A of $\mathbf{A}$, the relation symbol D_2 is interpreted by the universe B of $\mathbf{B}$, and the remaining relation and constant symbols of $\sigma_1 + \sigma_2$ are interpreted by the corresponding relations and constants of $\mathbf{A}$ and $\mathbf{B}$. This encoding makes it possible to formally view queries on pairs of σ-structures as queries on single $\sigma_1 + \sigma_2$-structures.

Note that FO-equivalence coincides with the Isomorphism Problem, since the isomorphism type of every finite σ-structure is FO-definable; as a result, FO-equivalence is not FO-definable. The same line of reasoning shows that $\mathrm{L}^\omega_{\infty\omega}$-equivalence is not $\mathrm{L}^\omega_{\infty\omega}$-definable. In what follows, we will investigate the logical definability and computational complexity of $\mathrm{L}^k_{\infty\omega}$-equivalence, $k \geq 1$.

Using Theorem 7.9 and the infinitary syntax of $\mathrm{L}^k_{\infty\omega}$, it is easy to see that $\mathrm{L}^k_{\infty\omega}$-equivalence is $\mathrm{L}^k_{\infty\omega}$-definable, $k \geq 1$. Indeed, as in the proof of Theorem 7.10, for every finite σ-structure $\mathbf{A}$, let $\Psi_{\mathbf{A}}$ be the conjunction of all $\mathrm{FO}^k_{\omega\omega}$-sentences satisfied by $\mathbf{A}$; clearly, $\Psi_{\mathbf{A}}$ is an $\mathrm{L}^k_{\infty\omega}$-sentence. Note that for every $\mathrm{L}^k_{\infty\omega}$-sentence Ψ, there are $\mathrm{L}^k_{\infty\omega}$-sentences Ψ^1 and Ψ^2 over $\sigma_1 + \sigma_2$ such that for all σ-structures $\mathbf{A}$ and $\mathbf{B}$, the following hold:

- $\mathbf{A} + \mathbf{B} \models \Psi_1$ if and only if $\mathbf{A} \models \Psi$.
- $\mathbf{A} + \mathbf{B} \models \Psi_2$ if and only if $\mathbf{B} \models \Psi$.

Finally, let $\mathbf{A}_1, \ldots, \mathbf{A}_n, \ldots$ be a list of representatives of all isomorphism types of finite σ-structures. Then $\mathrm{L}^k_{\infty\omega}$-equivalence on σ is definable by the $\mathrm{L}^k_{\infty\omega}$-sentence

$$\bigvee\{((\Psi^1_{\mathbf{A}_i} \wedge \Psi^2_{\mathbf{A}_j}) \rightarrow \Psi^1_{\mathbf{A_j}}) : i \geq 1, j \geq 1\}.$$

The preceding construction shows that $\mathrm{L}^k_{\infty\omega}$ is powerful enough to express its own equivalence, but provides no information about the computational complexity of $\mathrm{L}^k_{\infty\omega}$-equivalence. Nonetheless, the characterization of $\mathrm{L}^k_{\infty\omega}$-equivalence in terms of k-pebble games can be used to show that $\mathrm{L}^k_{\infty\omega}$-equivalance if LFP-definable and, thus, it is also polynomial-time computable. This result, whose proof is outlined next, was obtained by Dawar, Lindell and Weinstein [DLW95] and by Kolaitis and Vardi [KV92a] independently.

Proposition 7.18: *Let σ be a vocabulary and k a positive integer. Then there is a positive first-order formula $\varphi(x_1, \ldots, x_k, y_1, \ldots, y_k, S)$ over the vocabulary $\sigma_1 + \sigma_2$ such that the least fixed-point $\varphi^\infty(x_1, \ldots, x_k, y_1, \ldots, y_k)$ of this formula defines the query: "given two σ-structures $\mathbf{A}$, $\mathbf{B}$ and two k-tuples $(a_1 \ldots, a_k) \in A^k$ and $(b_1, \ldots, b_k) \in B^k$, is $(\mathbf{A}, a_1, \ldots, a_k) \not\equiv^k_{\infty\omega} (\mathbf{B}, b_1 \ldots, b_k)$?"*

Consequently, for each $k \geq 2$, $\mathrm{L}^k_{\infty\omega}$-equivalence is LFP*-definable.*

Proof: From the proof of Theorem 7.9, it follows that $(\mathbf{A}, a_1, \ldots, a_k) \not\equiv^k_{\infty\omega} (\mathbf{A}, b_1 \ldots, b_k)$ if and only if the Spoiler wins the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$ starting with the *configuration* $(a_1, \ldots, a_k, b_1, \ldots, b_k)$, that is, the Spoiler wins the k-pebble when the game begins with pebbles of the same label placed on a_i and b_i, $i = 1, \ldots, k$. The latter statement is definable by the least fixed-point φ^∞ of a positive first-order formula $\varphi(x_1, \ldots, x_k, y_1, \ldots, y_k, S)$ over the vocabulary $\sigma_1 + \sigma_2$ with a total of $2k$ distinct variables, which, intuitively, asserts that the Spoiler wins in the initial configuration or he wins in the "next" move of the game. More precisely, $\varphi(x_1, \ldots, x_k, y_1, \ldots, y_k S)$ is the formula

$$\chi(x_1, \ldots, x_k, y_1, \ldots, y_k) \vee (\vee^k_{i=1} \psi_i(x_1, \ldots, x_k, y_1, \ldots, y_k, S)),$$

where

- χ is a quantifier-free formula stating that $x_i \in D_1$, $y_i, \in D_2$, for $i = 1, \ldots, k$, and the substructures generated by $\{x_1, \ldots, x_k\}$ and $\{y_1, \ldots, y_k\}$ are not isomorphic;
- ψ_i is the formula

$$(\exists x_i \in D_1)(\forall y_i \in D_2) S(x_1, \ldots, x_k, y_1, \ldots, y_k) \vee (\exists y_i \in D_2)(\forall x_i \in D_1) S(x_1, \ldots, x_k, y_1, \ldots, y_k). \blacksquare$$

Proposition 7.18 implies that for each $k \geq 2$, $\mathrm{L}^k_{\infty\omega}$-equivalence is in P. Grohe [Gro99] established the following matching lower bound for the computational complexity of $\mathrm{L}^k_{\infty\omega}$-equivalence.

Theorem 7.19: *Let σ be a vocabulary containing at least one binary relation symbol. For each positive integer $k \geq 2$, the following problem is* P*-complete: given two finite σ-structures $\mathbf{A}$ and $\mathbf{B}$, does the Duplicator win the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$?*

Consequently, for each $k \geq 2$, $\mathrm{L}^k_{\infty\omega}$-equivalence is P*-complete.*

This result is proved via an intricate reduction from the MONOTONE CIRCUIT VALUE PROBLEM. Note that it provides a sharp contrast between the k-pebble game and the r-move Ehrenfeucht-Fraïssé-game, since, by Theorem 4.4, for each fixed $r \geq 1$, determining the winner in the r-move Ehrenfeucht-Fraïssé-game is solvable in logarithmic space (hence, it is unlikely to be P-complete). Note that if the number k of pebbles is also part of the input, then determining the winner in the k-pebble game is solvable in exponential time. It has been conjectured, but it has not been proved, that this upper bound is tight, which means that the following query is EXPTIME-complete: given a positive integer k and two finite σ-structures $\mathbf{A}$ and $\mathbf{B}$, does the Duplicator win the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$? This would complement Theorem 4.5 by Pezzoli to the effect that, when the number r of moves is part of the input, determining the winner in the r-move Ehrenfeucht-Fraïssé-game is a PSPACE-complete problem.

For every finite σ-structure $\mathbf{B}$, let $[\mathbf{B}]^k_{\infty\omega}$ be the $\mathrm{L}^k_{\infty\omega}$-equivalence class of $\mathbf{B}$ on finite σ-structures, that is,

$$[\mathbf{B}]^k_{\infty\omega} = \{\mathbf{A} \in \mathcal{F} : \mathbf{A} \equiv^k_{\infty\omega} \mathbf{B}\}.$$

Clearly, $[\mathbf{B}]^k_{\infty\omega}$ can also be viewed as a Boolean query $Q^k_{\mathbf{B}}$ on the class $\mathcal{F}$ of all finite σ-structures: given a finite σ-structure $\mathbf{A}$, is $\mathbf{A} \equiv^k_{\infty\omega} \mathbf{B}$? For every finite σ-structure $\mathbf{B}$ and every k-tuple $\mathbf{b}$ from B, we can also consider the related k-ary query $Q^k_{\mathbf{B},\mathbf{b}}$ on $\mathcal{F}$ such that, given a finite σ-structure $\mathbf{A}$, we have that

$$Q^k_{\mathbf{B},\mathbf{b}}(\mathbf{A}) = \{\mathbf{a} \in A^k : (\mathbf{A},\mathbf{a}) \equiv^k_{\infty\omega} (\mathbf{B},\mathbf{b})\}.$$

Theorem 7.9 implies that the query $Q^k_{\mathbf{B}}$ is definable by the $\mathrm{FO}^k_{\omega\omega}$-sentence $\bigwedge \Psi_{\mathbf{B}}$, where, as earlier, $\Psi_{\mathbf{B}}$ is the set of all $\mathrm{FO}^k_{\omega\omega}$-sentences satisfied by $\mathbf{B}$. Similarly, each query $Q^k_{\mathbf{B},\mathbf{b}}$ is $\mathrm{L}^k_{\infty\omega}$-definable as well. Dawar, Lindell and Weinstein [DLW95] established a much stronger result by showing that all queries $Q^k_{\mathbf{B}}$ and $Q^k_{\mathbf{B},\mathbf{b}}$ are actually $\mathrm{FO}^k_{\omega\omega}$-definable. This was achieved via a careful adaptation to $\mathrm{L}^k_{\infty\omega}$ of Scott's [Sco65] theorem to the effect that the isomorphism type of every countable structure is definable in the infinitary logic $\mathrm{L}_{\omega_1\omega}$. Here, we outline a different proof that was given in [KV96]. As a stepping stone, we first establish the following result.

Proposition 7.20: *Let $\mathbf{B}$ be a finite σ-structure and let $\mathbf{b}_1, \ldots, \mathbf{b}_l$ be an enumeration of all k-tuples from $\mathbf{B}$. For every positive integer k, there is a system $\mathsf{S}_{\mathbf{B}} = (\varphi_{\mathbf{B},\mathbf{b}_i}(x_1, \ldots, x_k, T_{\mathbf{b}_1}, \ldots, T_{\mathbf{b}_l}), 1 \leq i \leq l)$ of $\mathrm{FO}^k_{\omega\omega}$-formulas that are positive in $T_{\mathbf{b}_1}, \ldots, T_{\mathbf{b}_l}$ and have the property that $\varphi^{\infty}_{\mathbf{B},\mathbf{b}_i}(x_1, \ldots, x_k)$ defines the complement of the query $Q^k_{\mathbf{B},\mathbf{b}_i}$, $1 \leq i \leq l$. Thus, for every finite σ-structure $\mathbf{A}$ and every k-tuple $\mathbf{a}$ from $\mathbf{A}$*

$$(\mathbf{A},\mathbf{a}) \equiv^k_{\infty\omega} (\mathbf{B},\mathbf{b}) \iff \mathbf{A},\mathbf{a} \models \neg\varphi^{\infty}_{\mathbf{B},\mathbf{b}_i}(x_1, \ldots, x_k).$$

Proof: *(Outline)* For every $i \leq l$, let $\chi_{\mathbf{B},\mathbf{b}_i}(x_1, \ldots, x_k)$ be the conjunction of all atomic or negated atomic formulas $\eta(x_1, \ldots, x_k)$ such that $\mathbf{B}, \mathbf{b}_i \models \eta(x_1, \ldots, x_k)$. Moreover, for every j such that $1 \leq j \leq k$ and every element b from the universe of $\mathbf{B}$, we let $\mathbf{b}_i[j/b]$ be the k-tuple obtained from the k-tuple $\mathbf{b}_i = (b^i_1, \ldots, b^i_k)$ by replacing b^i_j by b. We then consider the system $\mathsf{S}_{\mathbf{B}} = (\varphi_{\mathbf{B},\mathbf{b}_i}$, $1 \leq i \leq l)$, where $\varphi_{\mathbf{B},\mathbf{b}_i}(x_1, \ldots, x_k, T_{\mathbf{b}_1} \ldots, T_{\mathbf{b}_l})$ is the formula

$$\neg\chi_{\mathbf{B},\mathbf{b}_i}(x_1, \ldots, x_k) \vee [\bigvee_{j=1}^{k} (\exists x_j) \bigwedge_{b \in B} T_{\mathbf{b}_i[j/b]}(x_1, \ldots, x_k)] \vee [\bigvee_{j=1}^{k} \bigvee_{b \in B} (\forall x_j) T_{\mathbf{b}_i[j/b]}(x_1, \ldots, x_k)].$$

By induction on m simultaneously for all $i \leq l$, it can be shown that on every σ-structure $\mathbf{A}$ the component $\Phi^m_{\mathbf{B},\mathbf{b}_i}$, $1 \leq i \leq l$, of the m-th stage of the system $\mathsf{S}_{\mathbf{B}}$ consists of all k-tuples $\mathbf{a}$ from $\mathbf{A}$ such that the Spoiler can win within m rounds the k-pebble game on $(\mathbf{A},\mathbf{a})$ and $(\mathbf{B},\mathbf{b})$. ∎

By combining Theorem 7.2 with Proposition 7.20, we can now obtain the result by Dawar, Lindell and Weinstein [DLW95] to the effect that every $\equiv^k_{\infty\omega}$-equivalence class is $\mathrm{FO}^k_{\omega\omega}$-definable.

Theorem 7.21: *Let k be a positive integer, $\mathbf{B}$ a finite σ-structure, and $\mathbf{b}$ a k-tuple from $\mathbf{B}$.*

- *The k-ary query $Q^k_{\mathbf{B},\mathbf{b}}$ is definable by some $\mathrm{FO}^k_{\omega\omega}$-formula $\theta_{\mathbf{B},\mathbf{b}}(x_1,\ldots,x_k)$.*
- *The Boolean query $Q^k_{\mathbf{B}}$ is definable by some $\mathrm{FO}^k_{\omega\omega}$-sentence $\theta_{\mathbf{B}}$ of FO^k.*

  In other words, for each fixed positive integer k and each fixed finite σ-structure $\mathbf{B}$, the following query is FO^k-definable: "Given a finite σ-structure $\mathbf{A}$, does the Duplicator win the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$?"

Proof: *(Outline)* Let $\mathbf{b}_1,\ldots,\mathbf{b}_l$ be an enumeration of all k-tuples from the universe of $\mathbf{B}$ and let $\mathsf{S}_{\mathbf{B}} = (\varphi_{\mathbf{B},\mathbf{b}_i}, 1 \leq i \leq l)$, be the system of positive FO^k-formulas used in the proof of Theorem 7.20. Theorem 7.2 implies that for every $i \leq l$ and every $m \geq 1$, there is a $\mathrm{FO}^k_{\omega\omega}$-formula $\varphi^m_{\mathbf{B},\mathbf{b}_i}(x_1,\ldots,x_k)$ that defines the component $\Phi^m_{\mathbf{B},\mathbf{b}_i}$ of the m-th stage of this system.

Let us now apply the system $\mathsf{S}_{\mathbf{B}}$ to the structure $\mathbf{B}$ itself. Then there is a positive integer m_0 such that on $\mathbf{B}$ the least fixed-point of this system is equal to its m_0-th stage, that is, for every $i \leq l$

$$\mathbf{B} \models (\forall x_1 \ldots \forall x_k)[\varphi^{m_0}_{\mathbf{B},\mathbf{b}_i}(x_1,\ldots,x_k) \leftrightarrow \varphi^{m_0+1}_{\mathbf{B},\mathbf{b}_i}(x_1,\ldots,x_k)].$$

It can then be shown that the query $Q^k_{\mathbf{B},\mathbf{b}}$ is definable by the following $\mathrm{FO}^k_{\omega\omega}$-formula $\theta_{\mathbf{B},\mathbf{b}}(x_1,\ldots,x_k)$:

$$\neg\varphi^{m_0}_{\mathbf{B},\mathbf{b}}(x_1,\ldots,x_k) \wedge [\bigwedge_{i=1}^{l}(\forall x_1 \ldots \forall x_k)(\varphi^{m_0}_{\mathbf{B},\mathbf{b}_i}(x_1,\ldots,x_k) \leftrightarrow \varphi^{m_0+1}_{\mathbf{B},\mathbf{b}_i}(x_1,\ldots,x_k))].$$

Finally, the query $Q^k_{\mathbf{B}}$ is definable by the $\mathrm{FO}^k_{\omega\omega}$-sentence $(\exists x_1 \ldots \exists x_k)(\bigvee_{i=1}^{l} \theta_{\mathbf{B},\mathbf{b}_i}(x_1,\ldots,x_k))$. ∎

The preceding Theorem 7.21 yields the following normal form for $\mathrm{L}^k_{\infty\omega}$-definability on finite structures, $k \geq 1$, a result due to Dawar, Lindell and Weinstein [DLW95].

Corollary 7.22: *Let σ be a vocabulary and k a positive integer. For every $\mathrm{L}^k_{\infty\omega}$-sentence ψ, there are $\mathrm{FO}^k_{\omega\omega}$-sentences ψ_m, $m \geq 1$, such that for every finite σ-structure $\mathbf{A}$, we have that*

$$\mathbf{A} \models \psi \iff \mathbf{A} \models \bigvee_{i=1}^{\infty} \psi_m.$$

Proof: The class of finite σ-structures that satisfy ψ is equal to the union of all $\equiv^k_{\infty\omega}$-equivalence classes of finite σ-structures that satisfy ψ. Thus, the desired sentences ψ_m are the $\mathrm{FO}^k_{\omega\omega}$-sentences $\theta_{\mathbf{B}}$, where $\mathbf{B}$ varies over all finite σ-structures that satisfy ψ. ∎

Since $\mathrm{L}^k_{\infty\omega}$ and $\mathrm{FO}^k_{\omega\omega}$ are closed under negations, we also have that on the class of all finite σ-structures, every $\mathrm{L}^k_{\infty\omega}$-sentence is equivalent to a countable conjunction of $\mathrm{FO}^k_{\omega\omega}$-sentences. Thus, that the expressive power of $\mathrm{L}^k_{\infty\omega}$ on finite structures reduces to a single application of infinitary disjunction or infinitary conjunction to a countable set of $\mathrm{FO}^k_{\omega\omega}$-sentences.

7.5 Least Fixed-Point Logic vs. Partial Fixed-Point Logic on Finite Structures

We now take a closer look at the relationship between least fixed-point logic LFP and partial fixed-point logic PFP on finite structures. Since every LFP-formula is also a PFP-formula, we have that on the class $\mathcal{F}$ of all finite σ-structures, $\mathrm{LFP}(\mathcal{F}) \subseteq \mathrm{PFP}(\mathcal{F})$. Recall that Theorem 6.11 asserts that on the class $\mathcal{O}$ of all ordered finite σ-structures, we have that $\mathrm{LFP}(\mathcal{O}) = \mathrm{P}(\mathcal{O})$; moreover, Theorem 6.38 asserts that $\mathrm{PFP}(\mathcal{O}) = \mathrm{PSPACE}(\mathcal{O})$. Consequently, $\mathrm{LFP}(\mathcal{O}) \neq \mathrm{PFP}(\mathcal{O})$ if and only if $\mathrm{P} \neq \mathrm{PSPACE}$. Thus, showing that PFP has strictly higher expressive power than LFP on the class $\mathcal{O}$ of all ordered finite σ-structures amounts to resolving one of the outstanding open problems in computational complexity. Chandra and Harel [CH82] raised the question of how LFP and PFP compare in terms of expressive power on $\mathcal{F}$ and conjectured that $\mathrm{LFP}(\mathcal{F}) \neq \mathrm{PFP}(\mathcal{F})$. Initially, researchers in finite model theory speculated that this conjecture is not equivalent to some open problem in complexity theory; moreover, they felt that it would be possible to confirm it using existing techniques. To justify this intuition, recall that, by Fagin's Theorem 4.7, ESO = NP on every class $\mathcal{C}$ of finite σ-structures. Consequently, showing that $\mathrm{LFP}(\mathcal{O}) \neq \mathrm{ESO}(\mathcal{O})$ amounts to establishing that $\mathrm{P} \neq \mathrm{NP}$. In contrast, $\mathrm{LFP}(\mathcal{F}) \neq \mathrm{ESO}(\mathcal{F})$, since, as shown in this section, the Even Cardinality query is not LFP-definable on $\mathcal{F}$, but, of course, it is ESO-definable on $\mathcal{F}$. Similarly, $\mathrm{PFP}(\mathcal{F}) \neq \mathrm{ESO}(\mathcal{F})$, since the Even Cardinality query is not PFP-definable on $\mathcal{F}$. So, it seems plausible that one could separate LFP from PFP on $\mathcal{F}$ by introducing suitable combinatorial games that would make it possible to differentiate between these two fixed-point logics on $\mathcal{F}$; of course, these games would have to be different from the k-pebble games, $k \geq 1$, since k-pebble games capture definability in the finite variable infinitary logic $\mathrm{L}^{\omega}_{\infty\omega}$, which subsumes both LFP and PFP on classes of finite structures. It turned out, however, that this intution was wrong. Indeed, a decade after Chandra and Harel [CH82] formulated their conjecture, Abiteboul and Vianu [AV91b] established that the separation of LFP from PFP on the class F of all finite σ-structures is literally equivalent to the separation of P from PSPACE. In what follows, we will highlight some of the key ideas that go into the proof of this result. For a complete proof, we refer the reader to the paper by Abiteboul and Vianu [AV91b] and to the subsequent excellent exposition by Dawar, Lindell and Weinstein [DLW95].

Definition 7.23: Let σ be a vocabulary and k a positive integer.

- If $\mathbf{A}$ is a finite σ-structure and $\mathbf{a}$ is a k-tuple of element of $\mathbf{A}$, then the *k-type of $\mathbf{a}$ on $\mathbf{A}$* is the collection of all $\mathrm{L}^{k}_{\infty\omega}$-formulas $\varphi(\mathbf{x})$ such that $\mathbf{A} \models \varphi(\mathbf{a})$.
- If $\mathbf{A}$ is a finite σ-structure and $\mathbf{a}$, $\mathbf{b}$ are two k-tuples of element of $\mathbf{A}$, then we write $\mathbf{A} \equiv^{k,\mathbf{A}}_{\infty\omega} \mathbf{b}$ to denote that $\mathbf{a}$ and $\mathbf{b}$ have the same the k-type on $\mathbf{A}$.

As we have seen, LFP cannot express the Even Cardinality query on $\mathcal{F}$, but it can express every polynomial-time computable query on $\mathcal{O}$. It follows that no LFP-formula $\psi(x, y)$ exists such that for every finite σ-structure $\mathbf{A}$, this formula defines a linear order on the universe A of $\mathbf{A}$. In contrast, Abiteboul and Vianu [AV91b] showed that, for every $k \geq 1$, there is an LFP-formula such that, for every finite σ-structure $\mathbf{A}$, this formula defines (in a sense that has to be made precise) a linear order on the set of the equivalence classes of the equivalence relation $\equiv^{k,\mathbf{A}}_{\infty\omega}$.

By definition, a *linear preorder* on a set B is a binary relation $\preceq$ on B that is is reflexive, transitive, and has the property that for every b_1 and b_2 in B, we have that $b_1 \preceq b_2$ or $b_2 \preceq b_1$. Every linear preorder $\preceq$ gives rise to an equivalence relation $\equiv$ defined by the condition: $b_1 \equiv b_2$ if and only if $b_1 \preceq b_2$ or $b_2 \preceq b_1$. Moreover, $\preceq$ induces a linear order, also denoted by $\preceq$, on the quotient set $B/\equiv$ of the equivalence classes of $\equiv$, where $[b_1]_{\equiv} \preceq [b_2]_{\equiv}$ if and only if $b_1 \preceq b_2$.

The next theorem is the key technical result in Abiteboul and Vianu [AV91b]; we give a hint of a different proof that has been discovered by Dawar, Lindell and Weinstein [DLW95].

Theorem 7.24: *Let σ be a vocabulary and $\mathcal{F}$ the class of all finite σ-structures. For every positive integer k, there is an* LFP*-definable $2k$-ary query Q_k on $\mathcal{F}$ such that for every finite σ-structure $\mathbf{A}$, the value $Q_k(\mathbf{A})$ of this query on $\mathbf{A}$ is a linear preorder on A^k whose whose induced equivalence relation coincides with the equivalence relation $\equiv_{\infty\omega}^{k,\mathbf{A}}$ of k-types on $\mathbf{A}$.*

Proof: (*Hint)* Using the characterization of $\mathrm{L}_{\infty\omega}^k$-equivalence in terms of k-pebble games, it is possible to design a *color-refinement* algorithm such that on every finite σ-structure $\mathbf{A}$, it inductively preorders all k-tuples from $\mathbf{A}$ according to their k-type on $\mathbf{A}$. This algorithm is naturally expressed in inflationary fixed-point logic IFP, which, as shown by Gurevich and Shelah [GS86], has the same expressive power as LFP on the class of all finite σ-structures (see also E. Grädel's Chapter in this volume for additional information on IFP). ∎

We can finally present Abiteboul and Vianu's surprising resolution of Chandra and Harel's conjecture.

Theorem 7.25: ([AV91b]) *Let σ be a vocabulary and $\mathcal{F}$ the class of all finite σ-structures. Then the following statements are equivalent:*

1. $\mathrm{LFP}(\mathcal{F}) = \mathrm{PFP}(\mathcal{F})$.
2. $\mathrm{P} = \mathrm{PSPACE}$.

Proof: *(Hint)* Assume first that $\mathrm{LFP}(\mathcal{F}) = \mathrm{PFP}(\mathcal{F})$. As seen in Example 6.36, PFP can express PSPACE-complete queries on $\mathcal{F}$. Therefore, such queries are LFP-definable on $\mathcal{F}$. Since every LFP-definable query query is polynomial-time computable, it follows that $\mathrm{PSPACE} \subseteq \mathrm{P}$.

For the other direction, assume that $\mathrm{P} = \mathrm{PSPACE}$. We have to show that $\mathrm{LFP}(\mathcal{F}) = \mathrm{PFP}(\mathcal{F})$. This will require essentially all the machinery we have developed in this section. Fix a positive integer k. If $\mathbf{A}$ is a finite σ-structure, then the equivalence relation $\equiv_{\infty\omega}^{k,\mathbf{A}}$ induces a quotient structure $\mathbf{A}/\equiv_{\infty\omega}^{k,\mathbf{A}}$ with universe the equivalence classes $[\mathbf{a}]_{\equiv_{\infty\omega}^{\mathbf{k},\mathbf{A}}}$ of k-tuples from A. Let

$$\mathcal{F}/\equiv_{\infty\omega}^{k} = \{\mathbf{A}/\equiv_{\infty\omega}^{k,\mathbf{A}} : \mathbf{A} \in \mathcal{F}\}$$

be the class of all these quotient structures. Theorem 7.24 implies that there is an LFP-definable query that defines a linear order on the universe of every quotient structure $\mathbf{A}/\equiv_{\infty\omega}^{k,\mathbf{A}}$ in $\mathcal{F}/\equiv_{\infty\omega}^{k}$. Consequently, $\mathrm{LFP}(\mathcal{F}/\equiv_{\infty\omega}^{k}) = \mathrm{P}(\mathcal{F}/\equiv_{\infty\omega}^{k})$. We can now use *transfer properties* between $\mathcal{F}$ and $\mathcal{F}/\equiv_{\infty\omega}^{k}$ to show that $\mathrm{PFP}(\mathcal{F}) \subseteq \mathrm{LFP}(\mathcal{F})$, as indicated in the diagram below.

$$\begin{array}{ccc} \varphi \in \mathrm{PFP}(\mathcal{F}) & \equiv & \psi \in \mathrm{LFP}(\mathcal{F}) \\ \downarrow & & \uparrow \\ \varphi^* \in \mathrm{PFP}(\mathcal{F}/\equiv_{\infty\omega}^{k}) & \equiv & \psi^* \in \mathrm{LFP}(\mathcal{F}/\equiv_{\infty\omega}^{k}) \end{array}$$

Specifically, assume that Q is a query on $\mathcal{F}$ definable by a PFP-formula φ with k distinct variables. The formula φ can be "transformed" to a PFP-formula φ^* over the vocabulary of the quotient structures, so that φ^* defines the "transformation" of the query Q to a query Q^* on $\mathcal{F}/\equiv_{\infty\omega}^{k}$. Since Q^* is PFP-definable on $\mathcal{F}/\equiv_{\infty\omega}^{k}$, it is polynomial-space computable. The hypothesis $\mathrm{PSPACE} = \mathrm{P}$ implies that Q^* is polynomial-time computable, hence Q^* is LFP-definable

on $\mathcal{F}/\equiv^k_{\infty\omega}$. Let ψ^* be an LFP-formula that defines Q^* on $\mathcal{F}/\equiv^k_{\infty\omega}$. We can now "pull back" ψ^* and obtain an LFP-formula ψ that defines the query Q on $\mathcal{F}$. Thus, $\mathrm{PFP}(\mathcal{F}) \subseteq \mathrm{LFP}(\mathcal{F})$. ∎

Thus, the difference in computational power between polynomial-time and polynomial-space computations, if any, amounts to the difference in expressive power between least fixed-points and partial fixed-points of first-order formulas on the class of all finite structures.

8 Existential Infinitary Logics with Finitely Many Variables

In Section 7, we saw that the finite-variable infinitary logics $\mathrm{L}^k_{\infty\omega}$ and the k-pebble games, $k \geq 1$, provide powerful tools for analyzing the expressive power of least fixed-point logic LFP. Our goal in this section is to develop a similar methodology for analyzing the expressive power of the existential fragment of LFP and, in particular, the expressive power of Datalog and Datalog($\neq$). To this effect, we will introduce finite-variable existential infinitary logics and certain asymmetric pebble games that turn out to be tailored for the study of Datalog and Datalog($\neq$).

8.1 The infinitary logics $\exists\mathrm{L}^k_{\infty\omega}$ and $\exists\mathrm{L}^k_{\infty\omega}(\neq)$

Informally, an existential finite-variable infinitary logic is a fragment of $\mathrm{L}^\omega_{\infty\omega}$ in which the rules for constructing formulas do not include applications of universal quantification or negation. These fragments can be further differentiated depending on whether the basic formulas include negated equalities or negated atomic formulas. We now formally two of these fragments, originally introduced by Kolaitis and Vardi [KV95].

Definition 8.1: Let σ be a vocabulary.

- For every positive integer k, we write $\exists\mathrm{L}^k_{\infty\omega}$ to denote the collection of all $\mathrm{L}_{\infty\omega}$-formulas that have at most k distinct variables and are obtained from atomic formulas (which may be equality statements) using existential quantification, infinitary conjunction, and infinitary disjunction.

  We write $\exists\mathrm{FO}^k$ to denote the collection of all first-order $\exists\mathrm{L}^k_{\infty\omega}$-formulas.

- The finite-variable existential infinitary logic $\exists\mathrm{L}^\omega_{\infty\omega}$ is the union of all $\exists\mathrm{L}^k_{\infty\omega}$'s

$$\exists\mathrm{L}^\omega_{\infty\omega} = \bigcup_{k=1}^{\infty} \exists\mathrm{L}^k_{\infty\omega}.$$

- For every positive integer k, we write $\exists\mathrm{L}^k_{\infty\omega}(\neq)$ to denote the collection of all $\mathrm{L}_{\infty\omega}$-formulas that have at most k distinct variables and are obtained from atomic formulas and negated equality statements (that is, formulas of the form $t_1 \neq t_2$, where t_1, t_2 are among the k variables and the constant symbols of σ), using existential quantification, infinitary conjunction, and infinitary disjunction.

  We write $\exists\mathrm{FO}^k(\neq)$ to denote the collection of all first-order $\exists\mathrm{L}^k_{\infty\omega}(\neq)$-formulas.

- The finite-variable existential infinitary logic $\exists\mathrm{L}^\omega_{\infty\omega}(\neq)$ is the union of all $\exists\mathrm{L}^k_{\infty\omega}(\neq)$'s, that is,

$$\exists\mathrm{L}^\omega_{\infty\omega}(\neq) = \bigcup_{k=1}^{\infty} \exists\mathrm{L}^k_{\infty\omega}(\neq).$$

As an example, the expression

$$(\exists z)(E(x,z) \wedge (\exists x)(x = z \wedge (\exists z)(E(x,z) \wedge E(z,y))))$$

is an $\exists\mathrm{FO}^3$-formula that defines the query "there is a path of length 3 from x to y". Actually, for every $m \geq 1$, the query "there is a path of length m from x to y" is $\exists\mathrm{FO}^3$-definable. This is a special case of a result concerning the relationship between Datalog and $\exists\mathrm{L}^{\omega}_{\infty\omega}$. Before stating this result in precise terms, we need to introduce a parametrization of Datalog programs based on the number of variables occurring in the rules.

For every positive integer k, let k-Datalog be the collection of all Datalog programs in which the body of every rule has at most k distinct variables and also the head of every rule has at most k variables (the variables of the body may be different from the variables of the head). For instance, the Non-2-Colorability query is expressible in 4-Datalog, since, as seen Example 6.20, it is definable by the goal predicate Q of the Datalog program below, which asserts that the existence of a cycle of odd length:

$$\left|\begin{array}{lll} O(x,y) & :- & E(x,y) \\ O(x,y) & :- & E(x,z), E(z,w), O(z,y) \\ Q & :- & O(x,x) \end{array}\right.$$

A complete proof of the next result can be found in [KV00].

Theorem 8.2: *Let σ be a vocabulary, k a positive integer, and*

$$\varphi_1(x_1,\ldots,x_{n_1},S_1,\ldots,S_l),\ldots,\varphi_l(x_1,\ldots,x_{n_l},S_1,\ldots,S_l)$$

a system of positive $\exists\mathrm{FO}^k$-formulas over the vocabulary $\sigma \cup \{S_1,\ldots,S_l\}$. Then the following statements are true for the above system and for the operator Φ associated with it.

- *For every $m \geq 1$, each component Φ^m_i, $1 \leq i \leq l$, of the stage $\Phi^m = (\Phi^m_1,\ldots,\Phi^m_l)$ is $\exists\mathrm{FO}^k$-definable on all the class $\mathcal{S}$ of all σ-structures.*
- *Each component φ^∞_i, $1 \leq i \leq l$, of the least fixed-point $(\varphi^\infty_1,\ldots,\varphi^\infty_l)$ of the system is $\exists\mathrm{FO}^k$-definable on the class of all σ-structures.*

Consequently, every query definable by a k-Datalog program on the class $\mathcal{S}$ of all σ-structures is also is also $\exists\mathrm{L}^k_{\infty\omega}$-definable on $\mathcal{S}$. In symbols,

$$k\text{-Datalog}(\mathcal{S}) \ \subseteq \ \exists\mathrm{L}^k_{\infty\omega}(\mathcal{S}).$$

Proof: (*(Hint)* This result is proved by induction on m simultaneously for all $i \leq l$. As was the case with Theorem 7.2, the key idea is to reuse variables judiciously. Some additional technical difficulties arise from the limited syntax of $\exists\mathrm{L}^k_{\infty\omega}$. These are overcome by using the following closure property of $\exists\mathrm{FO}^k$-definable queries, which has to be established separately:

If Q is an $\exists\mathrm{FO}^k$-definable query and $\pi : \{1,\ldots,k\} \mapsto \{1,\ldots,k\}$ is a function, then the query Q_π is also $\exists\mathrm{FO}^k$-definable, where for every σ-structure A and every sequence $(a_1,\ldots,a_k)$ of elements from the universe of A

$$(a_1,\ldots,a_k) \in Q_\pi(A) \Longleftrightarrow (a_{\pi(1)},\ldots,a_{\pi(k)}) \in Q(A).$$

By refining the proof of Proposition 6.21, it can be shown that every query definable by a k-Datalog program is also definable by the least fixed-point of a system of positive $\exists\mathrm{FO}^k$-formulas. Consequently, k-Datalog$(\mathcal{S}) \subseteq \exists\mathrm{L}^k_{\infty\omega}(\mathcal{S})$. ∎

A result similar to Theorem 8.2 can be established about the relationship between k-Datalog$(\neq)$ and $\exists\mathrm{L}^\omega_{\infty\omega}(\neq)$.

Theorem 8.3: *Let k be a positive integer. Every query definable by a k-Datalog$(\neq)$ program on the class $\mathcal{S}$ of all σ-structures is also is also $\exists\mathrm{L}^k_{\infty\omega}(\neq)$-definable on $\mathcal{S}$.*

It should be pointed out that on the class $\mathcal{F}$ of all finite σ-structures structures k-Datalog is properly contained in $\exists\mathrm{L}^\omega_{\infty\omega}$, since the latter can express queries that are not computable. Similarly, k-Datalog$(\neq)$ is properly in $\exists\mathrm{L}^\omega_{\infty\omega}(\neq)$ on $\mathcal{F}$.

The preservation properties of Datalog and Datalog$(\neq)$ in Propositions 6.23 and 6.26 extend to $\exists\mathrm{L}^\omega_{\infty\omega}$ and $\exists\mathrm{L}^\omega_{\infty\omega}(\neq)$. Specifically, every $\exists\mathrm{L}^\omega_{\infty\omega}$-definable query is preserved under homomorphisms and every $\exists\mathrm{L}^\omega_{\infty\omega}$-definable query is preserved under one-to-one homomorphisms. These preservation properties give rise to sufficient, but not necessary, conditions for inexpressibility in $\exists\mathrm{L}^\omega_{\infty\omega}$ or in $\exists\mathrm{L}^\omega_{\infty\omega}(\neq)$. In what follows, we will introduce a variant of pebble games that can actually characterize definability in these two infinitary logics.

8.2 Existential Pebble Games

The k-pebble game is a symmetric game, in the sense that the Duplicator wins the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$ if and only if the Duplicator wins the k-pebble game on $\mathbf{B}$ and $\mathbf{A}$. This is a consequence of the following two properties of the k-pebble game:

1. In each move of the game, the Spoiler can choose one of the two structures and place or remove a pebble on that structure.

2. The payoff condition is that the substructures generated by the pebbled elements must be isomorphic.

Thus, we can reverse the order of $\mathbf{A}$ and $\mathbf{B}$ without affecting the winner of the k-pebble game. We are interested in games that can characterize definability in the k-variable existential infinitary logics $\exists\mathrm{L}^k_{\infty\omega}$, $k \geq 1$. A closer scrutiny of the relationship between k-pebble games and $\mathrm{L}^k_{\infty\omega}$ reveals that moves of the Spoiler on the structure $\mathbf{B}$ correspond to universal quantification in $\mathrm{L}^k_{\infty\omega}$-formulas. This suggests that games for $\exists\mathrm{L}^k_{\infty\omega}$ should be such that the Spoiler is limited to always playing on $\mathbf{A}$ (and the Duplicator is limited to always playing on $\mathbf{B}$). Moreover, the payoff condition should be modified appropriately to account for the absence of negation and universal quantification in $\exists\mathrm{L}^k_{\infty\omega}$. These considerations led to the introduction of *existential k-pebble games* in [KV00].

Definition 8.4: Let k be a positive integer, σ a vocabulary, and $\mathbf{A}$ and $\mathbf{B}$ two σ-steructures.

The $(\exists, k)$*-pebble game on* $\mathbf{A}$ *and* $\mathbf{B}$ is played between two players, called the *Spoiler* and the *Duplicator*, each of whom has k pebbles that are labeled $1, \ldots, k$. In each move, the Spoiler either places a pebble that is not currently used on an element of $\mathbf{A}$ or removes a pebble from an element of $\mathbf{A}$. The Duplicator responds by either placing the pebble with the same label on an element of $\mathbf{B}$ or by removing the pebble with the same label from an element of $\mathbf{B}$.

$$\begin{array}{rcccccc} \text{Spoiler plays on } \mathbf{A}: & a_1 & a_2 & \ldots & a_r & & \\ & \downarrow & \downarrow & \cdots & \downarrow & & \\ \text{Duplicator plays on } \mathbf{B}: & b_1 & b_2 & \ldots & b_r & & r \leq k \end{array}$$

Assume that at some point of time during the game, r pebbles have been placed on each structure, where $1 \leq r \leq k$, and let $(a_i, b_i) \in A \times B$, $1 \leq i \leq r$, be the pairs of elements of $\mathbf{A}$ and $\mathbf{B}$ such that the label of the pebble on a_i is the same as the label of the pebble on b_i. The *Spoiler wins the* $(\exists, k)$*-pebble game on* $\mathbf{A}$ *and* $\mathbf{B}$ at this point of time, if the mapping $a_i \mapsto b_i$, $1 \leq i \leq r$, is not an homomorphism between the substructures of $\mathbf{A}$ and $\mathbf{B}$ generated by $\{a_1, \ldots, a_r\}$ and $\{b_1, \ldots, b_r\}$, respectively.

The *Duplicator wins the* $(\exists, k)$*-pebble game on* $\mathbf{A}$ *and* $\mathbf{B}$ if the above never happens, which means that the Duplicator has a *winning strategy* that allows him to continue playing "forever" by maintaining a partial homomorphism at every point of time.

The $(\exists, \neq, k)$*-pebble game on* $\mathbf{A}$ *and* $\mathbf{B}$ is defined in an entirely analogous way with the exception that the payoff condition for the Duplicator is that the mapping $a_i \mapsto b_i$, $1 \leq i \leq r$, is a one-to-one homomorphism between the substructures of $\mathbf{A}$ and $\mathbf{B}$ generated by $\{a_1, \ldots, a_r\}$ and $\{b_1, \ldots, b_r\}$, respectively.

The concept of a winning strategy for the Duplicator in the $(\exists, k)$-pebble game and the $(\exists, \neq, k)$-pebble game can be made precise in terms of families of partial homomorphisms or partial one-to-one homomorphisms with appropriate closure and extension properties.

Definition 8.5: A *winning strategy for the Duplicator in the* $(\exists, k)$*-pebble game (respectively, in the* $(\exists, \neq, k)$*-pebble game) on* $\mathbf{A}$ *and* $\mathbf{B}$ is a nonempty family $\mathcal{I}$ of partial homomorphisms (respectively, partial one-to-one homomorphims) from $\mathbf{A}$ to $\mathbf{B}$ with the following properties.

1. If $f \in \mathcal{I}$, then $|f - \{(c_1^{\mathbf{A}}, c_1^{\mathbf{B}}) \ldots, (c_1^{\mathbf{A}}, c_s^{\mathbf{B}})\}| \leq k$.
2. $\mathcal{I}$ is *closed under subfunctions*:
   If $g \in \mathcal{I}$ and f is a function such that $\{(c_1,^{\mathbf{A}}, c_1^{\mathbf{B}}), \ldots, (c_s^{\mathbf{A}}, c_s^{\mathbf{B}})\} \subseteq f \subseteq g$, then $f \in \mathcal{I}$.
3. $\mathcal{I}$ has the *forth property up to* k:
   If $f \in \mathcal{I}$ and $|f - \{(c_1^{\mathbf{A}}, c_1^{\mathbf{B}}) \ldots, (c_1^{\mathbf{A}}, c_s^{\mathbf{B}})\}| < k$, then for every $a \in A$, there is $g \in \mathcal{I}$ so that $f \subseteq g$ and $a \in \mathrm{dom}(g)$.

It is clear that if the Duplicator wins the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$, then the Duplicator also wins the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$. The converse, however, is not always true. Intuitively, it is easier for the Duplicator to win the $(\exists, k)$-game than it is to win the k-pebble game, because the Spoiler can not switch between the two structures. Note also that, unlike the k-pebble game, the $(\exists, k)$-pebble game is asymmetric. For instance, the Spoiler wins the $(\exists, k+1)$-game on the cliques $\mathbf{K}_{k+1}$ and $\mathbf{K}_k$, but the Duplicator wins the $(\exists, k+1)$-game on the cliques $\mathbf{K}_k$ and $\mathbf{K}_{k+1}$. A similar state of affairs holds for the $(\exists, \neq, k)$-pebble game.

We now present the connection between existential pebble games and definability in the finite-variable existential infinitary logics.

Definition 8.6: Let k be a positive integer, and let $\mathbf{A}$ and $\mathbf{B}$ be two σ-structures.

- We write $\mathbf{A} \preceq^{\exists,k}_{\infty\omega} \mathbf{B}$ to denote that every $\exists \mathrm{L}^k_{\infty\omega}$-sentence that is true on $\mathbf{A}$ is also true on $\mathbf{B}$.
- We write $\mathbf{A} \preceq^{\exists,k}_{\omega\omega} \mathbf{B}$ to denote that every first-order sentence of $\exists \mathrm{L}^k_{\infty\omega}$ that is true on $\mathbf{A}$ is also true on $\mathbf{B}$.

- Let $a_1, \ldots, a_r$ be a sequence of elements from A and let $b_1, \ldots, b_r$ be a sequenec of elements from B, for some $r \leq k$.

  We write $(\mathbf{A}, a_1, \ldots, a_r) \preceq^{\exists,k}_{\infty\omega} (\mathbf{B}, b_1, \ldots, b_r)$ to denote that for every $\exists \mathrm{L}^k_{\infty\omega}$-formula $\varphi(v_1, \ldots, v_r)$ with free variables among $v_1, \ldots, v_r$, we have that

$$\mathbf{A} \models \varphi(v_1/a_1, \ldots, v_r/a_r) \Longrightarrow \mathbf{B} \models \varphi(v_1/a_1, \ldots, v_r/b_r).$$

The relation $\preceq^{\exists,\neq,k}_{\infty\omega}$ is defined in a similar manner with $\exists \mathrm{L}^k_{\infty\omega}(\neq)$ in place of $\exists \mathrm{L}^k_{\infty\omega}$.

Theorem 8.7: [KV95] *Let k be a positive integer, and let $\mathbf{A}$ and $\mathbf{B}$ be two σ-structures. Then the following statements are equivalent:*

- $\mathbf{A} \preceq^{\exists,k}_{\infty\omega} \mathbf{B}$.
- *The Duplicator wins the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$.*

Moreover, if $\mathbf{B}$ is finite, then the above statements are also equivalent to

- $\mathbf{A} \preceq^{\exists,k}_{\omega\omega} \mathbf{B}$.

A similar result holds for $\exists \mathrm{L}^k_{\infty\omega}(\neq)$ and the $(\exists, \neq, k)$-pebble game.

As a consequence of Theorem 8.7, we obtain a characterization of $\exists \mathrm{L}^\omega_{\infty\omega}$-definability on classes of finite structures.

Corollary 8.8: *Let σ be a vocabulary, $\mathcal{C}$ a class of finite σ-structures, and Q a Boolean query on $\mathcal{C}$. Then the following statements are equivalent:*

1. *Q is $\exists \mathrm{L}^\omega_{\infty\omega}$-definable on $\mathcal{C}$.*
2. *There is a positive integer k such that for every structure $\mathbf{A} \in \mathcal{C}$ and every structure $\mathbf{B} \in \mathcal{C}$, if $Q(\mathbf{A}) = 1$ and the Duplicator wins the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$, then $Q(\mathbf{B}) = 1$.*

Thus, we have a sound and complete method for studying $\exists \mathrm{L}^\omega_{\infty\omega}$-definability on classes of finite structures.

Method 8.9: The Method of $(\exists, k)$-Pebble Games for $\exists \mathrm{L}^\omega_{\infty\omega}$
Let σ be a vocaculary, $\mathcal{C}$ a class of finite σ-structures, and Q a Boolean query on $\mathcal{C}$.

Soundness: To show that Q is *not* $\exists \mathrm{L}^\omega_{\infty\omega}$-definable on $\mathcal{C}$, it suffices to show that for every positive integer k there are structures $\mathbf{A}_k$ and $\mathbf{B}_k$ in $\mathcal{C}$ such that

- $Q(\mathbf{A}_k) = 1$ and $Q(\mathbf{B}_k) = 0$.
- The Duplicator wins the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$.

Completeness: This method is also complete, that is, if Q is *not* $\exists \mathrm{L}^\omega_{\infty\omega}$-definable on $\mathcal{C}$, then for every positive integer k such structures $\mathbf{A}_k$ and $\mathbf{B}_k$ exist. ∎

A similar method can be used for studying $\exists L^{\omega}_{\infty\omega}(\neq)$-definability on classes of finite structures using $(\exists, \neq, k)$-pebble games, $k \geq 1$.

We now present some results concerning the descriptive and computational complexity of determining the winner in the $(\exists, k)$-pebble game, $k \geq 1$. These results should be compared with the results in Propositions 7.18 and 7.20 and in Theorem 7.21 about the descriptive and computational complexity of determining the winner in the k-pebble game, $k \geq 1$.

Theorem 8.10: ([KV00]) *Let σ be a vocabulary and let k be a positive integer.*

1. *The query: "Given two σ-structures $\mathbf{A}$ and $\mathbf{B}$, does the Spoiler win the $(\exists, k)$-pebble on $\mathbf{A}$ and $\mathbf{B}$?" is* LFP*-definable.*

   As a result, there is a polynomial-time algorithm such that, given two finite σ-structures $\mathbf{A}$ and $\mathbf{B}$, it determines whether the Spoiler wins the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$.

2. *For every finite σ-structure $\mathbf{B}$, there is a k-Datalog program $\rho_{\mathbf{B}}$ that expresses the query "Given a σ-structure $\mathbf{A}$, does the Spoiler win the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$?".*

Proof: *(Sketch)* For notational simplicity, let us assume that the vocabulary σ consists of relation symbols only. Let $\theta(x_1, \ldots, x_k, y_1, \ldots, y_k)$ be a quantifier-free formula over the vocabulary $\sigma_1 + \sigma_2$ asserting that the correspondence $x_i \mapsto y_i$, $1 \leq i \leq k$, is not a mapping or it is a mapping that is *not* a homomorphism from the substructure generated by $x_1, \ldots, x_k$ over the vocabulary σ_1 to the substructure induced by $y_1, \ldots, y_k$ over the vocabulary σ_2. In particular, θ is the disjunction of the following formulas:

- $x_i = x_j \wedge y_i \neq y_j$, for every $i, j \leq k$ such that $i \neq j$.
- $R_1(x_{i_1}, \ldots, x_{i_m}) \wedge \neg R_2(y_{i_1}, \ldots, y_{i_m})$, for every m-ary relation symbol R in σ and every m-ary tuple $(i_1, \ldots, i_m)$ of indices from the set $\{1, \ldots, k\}$.

Let T be a 2k-ary relation symbol not in $\sigma_1 + \sigma_2$ and let $\varphi(x_1, \ldots, x_k, y_1, \ldots, y_k, T)$ be the following positive first-order formula over the vocabulary $\sigma_1 + \sigma_2 \cup \{T\}$:

$$\theta(x_1, \ldots, x_k, y_1, \ldots, y_k) \vee \bigvee_{j=1}^{k} (\exists x_j \in D_1)(\forall y_j \in D_2) T(x_1, \ldots, x_k, y_1, \ldots, y_k).$$

It is easy to verify that if $\mathbf{A}$ and $\mathbf{B}$ are σ-structures and $(a_1, \ldots, a_k)$, $(b_1, \ldots, b_k)$ are k-tuples of elements from A and B respectively, then the following statements are equivalent:

1. $\mathbf{A} + \mathbf{B} \models \varphi^{\infty}(a_1, \ldots, a_k, b_1, \ldots, b_k)$.
2. The Spoiler wins the $(\exists, k)$-pebble game on $(A, a_1, \ldots, a_k)$ and $(B, b_1, \ldots, b_k)$.

Let ψ be the sentence $(\exists x_1) \cdots (\exists x_k)(\forall y_1) \cdots (\forall y_k) \varphi^{\infty}(x_1, \ldots, x_k, y_1, \ldots, y_k)$ of least fixed-point logic LFP. Consequently, for every σ-structure $\mathbf{A}$ and every σ-structure $\mathbf{B}$ the following statements are equivalent:

1. $\mathbf{A} + \mathbf{B} \models \psi$.
2. The Spoiler wins the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$.

Note that the positive first-order formula φ above involves existential quantifiers that are interpreted over the elements of $\mathbf{A}$, and universal quantifiers that are interpreted over the elements of $\mathbf{B}$. Consequently, if $\mathbf{B}$ is a fixed finite σ-structure, then the universal quantifiers can be replaced by finitary conjunctions over the elements of the universe B of $\mathbf{B}$ and, thus, φ can be transformed to a k-Datalog program ρ_p that expresses the query: "Given a finite σ-structure $\mathbf{A}$, does the Spoiler win the existential k-pebble game on $\mathbf{A}$ and $\mathbf{B}$?". In what follows, we describe this k-Datalog program in some detail. The goal of $\rho_{\mathbf{B}}$ is a 0-ary predicate S. Let $\mathbf{b} = (b_1, \ldots, b_k)$ be a k-tuple of elements of B. For each such k-tuple, we introduce a k-ary relation symbol $T_{\mathbf{b}}$ and the following rules:

- For every i and j such that $b_i \neq b_j$, we have a rule

$$T_{\mathbf{b}}(x'_1, \ldots, x'_k) \ :- \quad ,$$

  where $x'_i = x'_j = x_i$, and $x'_s = x_s$, for $s \neq i, j$.

- For every m-ary relation symbol R of σ and every m-ary tuple $(i_1, \ldots, i_m)$ such that

$$B, b_{i_1}, \ldots, b_{i_m} \models \neg R(x_{i_1}, \ldots, x_{i_m}),$$

  we have a rule

$$T_{\mathbf{b}}(x_1, \ldots, x_k) \ :- \ R(x_{i_1}, \ldots, x_{i_m}).$$

- For every j with $1 \leq j \leq k$, we have a rule

$$T(x_1, \ldots, x_k) \ :- \ \bigwedge_{c \in B} T_{\mathbf{b}[j/c]}(x_1, \ldots, x_{j-1}, y, x_{j+1}, \ldots, x_k),$$

  where $\mathbf{b}[j/c] = (b_1, \ldots, b_{j-1}, c, b_{j+1}, \ldots, b_k)$ and y is a new variable (note, however, that the body of the rule has k variables).

- For the goal predicate S, we have the rule

$$S \ :- \ \bigwedge_{\mathbf{b} \in B^k} T_{\mathbf{b}}(x_1, \ldots, x_k). \quad \blacksquare$$

As stated in Theorem 7.19, Grohe [Gro99] showed that if σ is a vocabulary containing at least one binary relation symbol, then for every $k \geq 2$, the following query is P-complete: "given two finite σ-structures $\mathbf{A}$ and $\mathbf{B}$, does the Duplicator win the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$?". In this query both structures $\mathbf{A}$ and $\mathbf{B}$ are part of the input. Recall, however, that the complexity drops if the structure $\mathbf{B}$ is kept fixed. Indeed, as shown in Theorem 7.21, for each fixed positive integer k and for each fixed finite σ-structure $\mathbf{B}$, the following query is $\mathrm{FO}^k_{\omega\omega}$-definable (hence, solvable in logarithmic space): "given a finite σ-structure $\mathbf{A}$, does the Duplicator win the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$?". In contrast, we now show that determining the winner in the $(\exists, k)$-pebble game can be P-complete, even for a fixed k and a fixed $\mathbf{B}$.

Proposition 8.11: *There are a vocabulary σ consisting of relation symbols of arity at most 3 and a finite σ-structure $\mathbf{B}$ such that the following query is* P*-complete: "given a finite σ-structure $\mathbf{A}$, does the Duplicator win the $(\exists, 3)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$?".*

Proof: We will describe a logarithmic-space reduction from the satisfiability problem HORN 3-SAT for Horn formulas with at most three literals per clause, which is a well-known P-complete problem (see [GHR95])

Let σ be a vocabulary consisting of two unary relation symbols N_1 and P_1, two binary relation symbols N_2 and P_2, and two ternary relation symbols N_3 and P_3. The intuition is that these relation symbols represent the various types of clauses that may occur in a Horn formula with at most three literals per clause. Specifically, N_1 and P_1 will represent the unit clauses $\neg x$ and x, N_2 and P_2 will represent the binary Horn clauses $\neg x \vee \neg y$ and $\neg x \vee y$, while N_3 and P_3 will represent the ternary Horn clauses $\neg x \vee \neg y \vee \neg z$ and $\neg x \vee \neg y \vee z$. Let $\mathbf{B}$ be the Boolean σ-structure whose relations are the sets of satisfying truth assignments of Horn clauses with at most three literals per clause. More precisely, the universe of $\mathbf{B}$ is the set $\{0,1\}$ and the relations of $\mathbf{B}$ are as follows:

- $N_1^{\mathbf{B}} = \{0\}$ and $P_1^{\mathbf{B}} = \{1\}$;
- $N_2^{\mathbf{B}} = \{0,1\}^2 - \{(1,1)\}$ and $P_2^{\mathbf{B}} = \{0,1\}^2 - \{(1,0)\}$;
- $N_3^{\mathbf{B}} = \{0,1\}^3 - \{(1,1,1)\}$ and $N_3^{\mathbf{B}} = \{0,1\}^3 - \{(1,1,0)\}$.

If φ is a Horn formula with at most three literals per clause, then φ can be encoded by a finite σ-structure $\mathbf{A}_\varphi$ such that the universe A of $\mathbf{A}$ is the set of all variables occurring in φ and the relations on $\mathbf{A}$ represent the clauses of φ. For instance, $N_2^{\mathbf{A}}$ consists of all pairs (x,y) of variables such that $\neg x \vee \neg y$ is a clause of φ, while $P_3^{\mathbf{A}}$ consists of all triples (x,y,z) of variables such that $\neg x \vee \neg y \vee z$ is a clause of φ. Clearly, $\mathbf{A}_\varphi$ can be constructed in logarithmic space from φ.

We now claim that φ is satisfiable if and only if the Duplicator wins the $(\exists, 3)$-pebble game on $\mathbf{A}_\varphi$ and $\mathbf{B}$. If φ is satisfiable, then a satisfying truth assignment is a homomorphism from $\mathbf{A}_\varphi$ to $\mathbf{B}$. Hence, the Duplicator can win the $(\exists, 3)$-pebble game on $\mathbf{A}_\varphi$ and $\mathbf{B}$ by using the values of this homomorphism to respond to the moves of the Spoiler. In fact, in this case the Duplicator can win the $(\exists, k)$-pebble game on $\mathbf{A}_\varphi$ and $\mathbf{B}$ for every $k \geq 1$. The other direction requires more work. We start with the observation that the well-known polynomial-time *marking algorithm* for Horn satisfiability is readily expressible in 3-Datalog. More precisely, consider the following 3-Datalog program π with T and P as its IDB predicates and P as its goal predicate.

$$
\begin{array}{lcl}
T(z) & :- & P_1(z) \\
T(z) & :- & P_2(x,z), T(x) \\
T(z) & :- & P_3(x,y,z), T(x), T(y) \\
P & :- & N_1(x), T(x) \\
P & :- & N_2(x,y), T(x), T(y) \\
P & :- & N_3(x,y,z), T(x), T(y), T(z)
\end{array}
$$

It is easy to verify that $\mathbf{B}$ does not satisfy the goal predicate P. Moreover, a Horn formula φ with at most 3 literals per clause is unsatisfiable if and only if the structure $\mathbf{A}_\varphi$ satisfies the goal predicate P. This holds because the first three rules of π mimick the marking algorithm for Horn satisfiability by putting into the predicate T all variables of φ that must take value "true" in every satisfying truth assignment; the last three rules capture the possible ways in which a Horn formula is found to be unsatisfiable by this algorithm because all variables occurring in some negative clause are forced to take value "true". Assume now that the Duplicator wins the $(\exists, 3)$-pebble game on $\mathbf{A}_\varphi$ and $\mathbf{B}$. We claim that φ is satisfiable. If this is not the case, then $\mathbf{A}_\varphi$ satisfies the goal predicate P of the above 3-Datalog program π. Since the Duplicator wins the $(\exists, 3)$-pebble game on $\mathbf{A}_\varphi$ and $\mathbf{B}$, Theorem 8.8 implies that $\mathbf{B}$ satisfies the goal predicate P of π, which is not true. ∎

Obviously, the preceding Proposition 8.11 implies that, when both structures **A** and **B** are part of the input, then determining the winner in the $(\exists, 3)$-pebble game is a P-complete problem. In fact, it is known that this holds for every fixed $k \geq 2$ and for vocabularies consisting of a binary relation symbol and a fixed number of unary relation symbols [KP03]. As stated earlier, it has been conjectured, but remains to be proved, that determining the winner in the k-pebble game with k part of the input is an EXPTIME-complete problem. In contrast, determining the winner in the $(\exists, k)$-pebble game when k is part of the input has been shown to be EXPTIME-complete.

Theorem 8.12: ([KP03]) *The following problem is* EXPTIME*-complete: given a positive integer k, a vocabulary σ consisting of one binary relation symbol and a number of unary relation symbols, and two finite σ-structures* **A** *and* **B**, *does the Duplicator win the $(\exists, k)$-pebble game on* **A** *and* **B***?*

We note that some of the results about Datalog and $(\exists, k)$-pebble games that we presented here have found numerous applications to the study of constraint satisfaction problems, which is the topic of the Chapter by Moshe Y. Vardi in this volume.

8.3 Descriptive Complexity of Fixed Subgraph Homeomorphism Queries

The original motivation behind the introduction of $(\exists, k)$-pebble games and $(\exists, \neq, k)$-pebble games in [KV95] was to develop tools for analyzing the expressive power of Datalog and Datalog($\neq$). We now close this chapter by presenting a case study of the expressibility of certain important graph-theoretic problems in Datalog($\neq$) using $(\exists, \neq, k)$-pebble games.

Definition 8.13: Let **H** and **G** be two directed graphs.

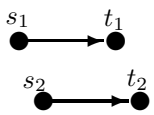
A *homeomorphism* $h : H \rightsquigarrow G$ from **H** to **G** is a one-to-one mapping from the nodes of **H** to the nodes of **G** such that h maps the edges of **H** to pairwise node-disjoint simple paths of **G**.

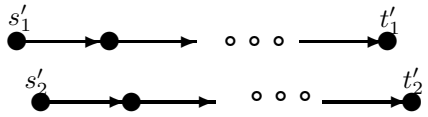
The concept of homeomorphism gives rise to a family of decision problems on directed graphs, one for each fixed finite directed graph **H**.

Definition 8.14: Let **H** be a fixed finite directed graph. The Fixed Subgraph Homeomorphism Query with Pattern **H**, denoted by FiSH(**H**), asks: given a directed graph **G** and a one-to-one mapping from the nodes of H to the nodes of G, is there a homeomorphism $h : H \rightsquigarrow G$ extending this mapping?

The following examples illustrate some typical members of this family of queries.

Example 8.15: Let **H** be a directed graph consisting of two parallel directed edges, that is, **H** has four nodes s_1, s_2, t_1, t_2 and two edges (s_1, t_1), (s_2, t_2).





Then FISH($\mathbf{H}$) is the 2-DISJOINT PATHS Query: given a directed graph $\mathbf{G}$ and four nodes s'_1, s'_2, t'_1, t'_2, does $\mathbf{G}$ contain two node-disjoint simple paths from s'_1 to t'_1 and from s'_2 to t'_2?

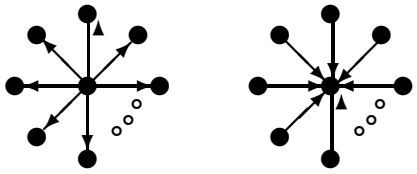
By taking $\mathbf{H}$ to be a graph consisting of m parallel directed edges, this example generalizes to the m-DISJOINT PATHS query, $m \geq 2$. ∎

Example 8.16: Let $\mathbf{C}_3$ be a directed cycle with three nodes. Then FISH($\mathbf{C}_3$) is the query: given a directed graph $\mathbf{G}$ and three nodes a_1, a_2, a_3, is there a simple cycle in $\mathbf{G}$ containing these nodes?

By taking $\mathbf{H}$ to be a directed cycle $\mathbf{C}_m$ with m nodes, $m \geq 3$, this example generalizes to the query: given a directed graph $\mathbf{G}$ and m nodes $a_1, \ldots, a_m$, is there a simple cycle in $\mathbf{G}$ containing these m nodes? ∎

Fortune, Hopcroft and Willey [FHW80] obtained a complete classification of the computational complexity of all FISH($\mathbf{H}$) queries, as $\mathbf{H}$ ranges over all finite directed graphs. Before stating this classification result, we need one more concept.

Definition 8.17: A **star** graph is a directed graph that consists either of a single *source* node and edges emanating from this node or of a single *sink* node with edges terminating on this node.



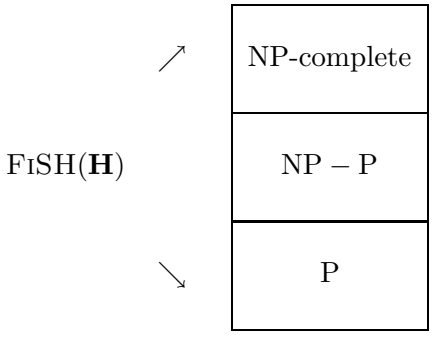
Theorem 8.18: *The following dichotomy holds for the computational complexity of the* FIXED SUBGRAPH HOMEOMORPHISM QUERY WITH PATTERN $\mathbf{H}$*, where* $\mathbf{H}$ *ranges over all finite directed graphs.*

- *If* $\mathbf{H}$ *is a star graph, then* FISH($\mathbf{H}$) *is in* P.
- *If* $\mathbf{H}$ *is not a star graph, then* FISH($\mathbf{H}$) *is* NP*-complete.*

Let us digress for moment and explain why the preceding result is a dichotomy theorem. Ladner [Lad75] showed that if P ≠ NP, then there is a decision problem Q such that

- $Q \in$ NP − P;
- Q is *not* NP-complete.

Thus, if P ≠ NP, then NP contains problems of intermediate complexity between polynomial-time solvability and NP-completeness. Theorem 8.18, however, asserts that no FISH($\mathbf{H}$) query is a problem of such intermediate complexity.



Note that the dichotomy in the computational complexity of FISH($\mathbf{H}$) queries is proper only if P $\neq$ NP. We now present a dichotomy in the descriptive complexity of FISH($\mathbf{H}$) queries that does not depend on any complexity-theoretic assumptions.

Theorem 8.19: ([KV95]) *The following dichotomy holds for the descriptive complexity of the* FIXED SUBGRAPH HOMEOMORPHISM QUERY WITH PATTERN $\mathbf{H}$, *where* $\mathbf{H}$ *ranges over all finite directed graphs.*

- *If* $\mathbf{H}$ *is a star graph, then* FISH($\mathbf{H}$) *is* $\exists L^{\omega}_{\infty\omega}(\neq)$*-definable; in fact, it is definable in Datalog(*$\neq$*).*
- *If* $\mathbf{H}$ *is not a star graph, then* FISH($\mathbf{H}$) *is not* $\exists L^{\omega}_{\infty\omega}(\neq)$*-definable.*

Proof: *(Hint)* If $\mathbf{H}$ is a star graph, then the FISH($\mathbf{H}$) query is solvable in polynomial time using a max flow algorithm, which can be expressed by a Datalog($\neq$) program.

The 2-DISJOINT PATHS query is the key case of the results concerning the inexpressibility of FISH($\mathbf{H}$) in Datalog($\neq$) when $\mathbf{H}$ is not a star graph. To this effect, one can show that for every $k \geq 1$ there are directed graphs $\mathbf{A}_k$ and $\mathbf{B}_k$ such that the following hold:

- $\mathbf{A}_k$ satisfies the 2-DISJOINT PATHS query, but $\mathbf{B}_k$ does not;
- the Duplicator wins the $(\exists, \neq, k)$-pebble game on $\mathbf{A}_k$, $\mathbf{B}_k$.

The graph $\mathbf{A}_k$ consists of two disjoint sufficiently long paths. The graph $\mathbf{B}_k$, however, is much more complicated and so is the description of the Duplicator's winning strategy in the $(\exists, \neq, k)$-pebble game on $\mathbf{A}_k$ and $\mathbf{B}_k$. This graph is extracted from the reduction of 3-SAT to 2-DISJOINT PATHS used by Fortune, Hopcroft and Willey [FHW80] to establish that the 2-DISJOINT PATHS query is NP-hard. ∎

Several remarks are in order now. The first is that the proof of Theorem 8.19 reveals that certain constructions used to prove NP-hardness can also be used to obtain interesting structures on which to play combinatorial logics and establish lower bounds for definability. Note that the dichotomy in the descriptive complexity of FISH($\mathbf{H}$) queries cannot be proved using preservation properties of Datalog($\neq$), because these queries are preserved under one-to-one homomorphisms.

It is an open problem to significantly strengthen the lower bound in Theorem 8.19 by establishing that if $\mathbf{H}$ is not a star graph, then the FISH($\mathbf{H}$) query is not $L^{\omega}_{\infty\omega}$-definable. The critical step would be to show that the 2-DISJOINT PATHS query is not $L^{\omega}_{\infty\omega}$-definable.

As a byproduct of their celebrated work on the graph minor problem, Robertson and Seymour [RS85, RS95] showed that every FISH($\mathbf{H}$) query is solvable in polynomial time when restricted to undirected graphs. It would be interesting to carry out a detailed study of the descriptive complexity of FISH($\mathbf{H}$) queries on undirected graphs. A preliminary investigation by Barland [Bar96] showed that the FISH($\mathbf{C}_3$) query is LFP-definable on the class $\mathcal{G}$ of all finite undirected graphs. This suggest that if there is a dichotomy in the descriptive complexity of FISH($\mathbf{H}$) queries on undirected graphs, then the boundary of that dichotomy is going to be different from the boundary of the dichotomy in Theorem 8.19.

9 References

References

[AF90] M. Ajtai and R. Fagin. Reachability is harder for directed than for undirected finite graphs. *Journal of Symbolic Logic*, 55(1):113–150, March 1990.

[AF97] S. Arora and R. Fagin. On winning strategies in Ehrenfeucht-Fraïssé games. *Theoretical Computer Science*, 174:97–121, 1997.

[AG87] M. Ajtai and Y. Gurevich. Monotone versus positive. *Journal of the ACM*, 34:1004–1015, 1987.

[AHV95] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of databases*. Addison-Wesley, 1995.

[AV91a] S. Abiteboul and V. Vianu. Datalog extensions for database queries and updates. *Journal of Computer and System Sciences*, 43:62–124, 1991.

[AV91b] S. Abiteboul and V. Vianu. Generic computation and its complexity. In *Proc. 23rd ACM Symp. on Theory of Computing*, pages 209–219, 1991.

[Bar77] J. Barwise. On Moschovakis closure ordinals. *Journal of Symbolic Logic*, 42:292–296, 1977.

[Bar96] I. Barland. *Expressing Optimization Problems as Integer Programs, and Undirected Path Problems: a Descriptive Complexity Approach*. PhD thesis, University of California, Santa Cruz, 1996.

[BGK85] A. Blass, Y. Gurevich, and D. Kozen. A zero–one law for logic with a fixed point operator. *Information and Control*, 67:70–90, 1985.

[CH80] A. Chandra and D. Harel. Computable queries for relational databases. *Journal of Computer and System Sciences*, 21:156–178, 1980.

[CH82] A. Chandra and D. Harel. Structure and complexity of relational queries. *Journal of Computer and System Sciences*, 25:99–128, 1982.

[CH85] A. Chandra and D. Harel. Horn clause queries and generalizations. *Journal of Logic Programming*, 1:1–15, 1985.

[Coo74] S. A. Cook. An observation of time-storage trade-off. *Journal of Computer and System Sciences*, 9:308–316, 1974.

[Daw98] A. Dawar. A restricted second-order logic for finite structures. *Information and Computation*, 143:154–174, 1998.

[Dic85] M. A. Dickmann. Larger infinitary languages. In J. Barwise and S. Feferman, editors, *Model-Theoretic Logics*, pages 317–363. Springer-Verlag, 1985.

[Die97] R. Diestel. *Graph Theory*. Springer, 1997.

[DLW95] A. Dawar, S. Lindell, and S. Weinstein. Infinitary logic and inductive definability over finite structures. *Information and Computation*, 119:160–175, 1995.

[dR87] M. de Rougemont. Second-order and inductive definability on finite structures. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 33:47–63, 1987.

[Ehr61] A. Ehrenfeucht. An application of games to the completeness problem for formalized theories. *Fund. Math.*, 49:129–141, 1961.

[End72] H. B. Enderton. *A Mathematical Introduction to Logic*. Academic Press, New York, 1972.

[Fag74] R. Fagin. Generalized first–order spectra and polynomial–time recognizable sets. In R. M. Karp, editor, *Complexity of Computation, SIAM-AMS Proceedings, Vol. 7*, pages 43–73, 1974.

[Fag75] R. Fagin. Monadic generalized spectra. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 21:89–96, 1975.

[Fag76] R. Fagin. Probabilities on finite models. *Journal of Symbolic Logic*, 41:50–58, 1976.

[Fag97] R. Fagin. Easier ways to win logical games. In N. Immerman and Ph. G. Kolaitis, editors, *Descriptive Complexity and Finite Models*, volume 31 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 1–32. American Mathematical Society, 1997.

[FHW80] S. Fortune, J. Hopcroft, and J. Wyllie. The directed homeomorphism problem. *Theoretical Computer Science*, 10:111–121, 1980.

[Fra54] R. Fraïssé. Sur quelques classifications des systèmes de relations. *Publ. Sci. Univ. Alger. Sér. A*, 1:35–182, 1954.

[FSV95] R. Fagin, L. Stockmeyer, and M. Y. Vardi. On monadic NP vs. monadic co-NP. *Information and Computation*, 120(1):78–92, July 1995.

[Gai82] H. Gaifman. On local and nonlocal properties. In J. Stern, editor, *Logic Colloquium '81*, pages 105–135. North Holland, 1982.

[GHR95] R. Greenlaw, H. J. Hoover, and W. L. Ruzzo. *Limits to parallel computation: P-completeness theory*. Oxford University Press, New York, 1995.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability - A Guide to the Theory of NP-Completeness*. W. H. Freeman and Co., 1979.

[GKLT69] Y. V. Glebskii, D. I. Kogan, M. I. Liogonki, and V. A. Talanov. Range and degree of realizability of formulas in the restricted predicate calculus. *Cybernetics*, 5:142–154, 1969.

[Gro95] M. Grohe. Complete problems for fixed–point logics. *Journal of Symbolic Logic*, 60(2):517–527, 1995.

[Gro99] M. Grohe. Equivalence in finite-variable logics is complete for polynomial time. *Combinatorica*, 19(4):507–523, 1999.

[GS86] Y. Gurevich and S. Shelah. Fixed-point extensions of first-order logic. *Annals of Pure and Applied Logic*, 32:265–280, 1986.

[GV85] H. Gaifman and M. Y. Vardi. A simple proof that connectivity is not first-order. *Bulletin of the European Association for Theoretical Computer Science*, 26:43–45, June 1985.

[Han65] W. Hanf. Model-theoretic methods in the study of elementary logic. In J. Addison, L. Henkin, and A. Tarski, editors, *The Theory of Models*, pages 132–145. North Holland, 1965.

[Har94] J Hartmanis. Turing Award Lecture: on computational complexity and the nature of computer science. *Communications of the ACM*, 37:37–43, 1994.

[Imm82] N. Immerman. Upper and lower bounds for first-order expressibility. *Journal of Computer and System Sciences*, 25:76–98, 1982.

[Imm86] N. Immerman. Relational queries computable in polynomial time. *Information and Control*, 68:86–104, 1986.

[Kei71] H. J. Keisler. *Model Theory for Infinitary Logic*. North Holland, 1971.

[Kle55] S. C. Kleene. Arithmetical predicates and function quantifiers. *Trans. Amer. Math. Soc.*, 79:312–340, 1955.

[Kna28] B. Knaster. Un theoreme sur les fonctions d'ensembles. *Ann. Soc. Polon. Math.*, 6:133–134, 1928.

[KP03] Ph.G. Kolaitis and J. Panttaja. On the complexity of existential pebble games. In *2003 Annual Conference of the European Association for Computer Science Logic - CSL '03*, Lecture Notes in Computer Science. Springer, 2003. to appear.

[KV87] Ph. G. Kolaitis and M. Y. Vardi. The decision problem for the probabilities of higher-order properties. In *Proc. 19th ACM Symp. on Theory of Computing*, pages 425–435, 1987.

[KV92a] Ph. G. Kolaitis and M. Y. Vardi. Fixpoint logic vs. infinitary logic in finite-model theory. In *Proc. 6th IEEE Symp. on Logic in Comp. Sci.*, pages 46–57, 1992.

[KV92b] Ph. G. Kolaitis and M. Y. Vardi. Infinitary logic and 0-1 laws. *Information and Computation*, 98:258–294, 1992. Special issue: Selections from the Fifth Annual IEEE Symposium on Logic in Computer Science.

[KV95] Ph. G. Kolaitis and M. Y. Vardi. On the expressive power of Datalog: tools and a case study. *Journal of Computer and System Sciences*, 51(1):110–134, August 1995. Special Issue: Selections from Ninth Annual ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS), Nashville, TN, USA, 2-4 April 1990.

[KV96] Ph.G. Kolaitis and M.Y. Vardi. On the expressive power of variable-confined logics. In *Proceedings of 11th Annual IEEE Symposium on Logic in Computer Science - LICS '96*, pages 348–59, 1996.

[KV00] Ph.G. Kolaitis and M.Y. Vardi. Conjunctive-query containment and constraint satisfaction. *Journal of Computer and System Sciences*, pages 302–332, 2000. Earlier version in: Proc. 17th ACM Symp. on Principles of Database Systems (PODS '98).

[Lad75] R. E. Ladner. On the structure of polynomial time reducibility. *Journal of the Association for Computing Machinery*, 22(1):155–171, 1975.

[McA95] M. McArthur. Convergence and 0-1 laws for $\mathrm{l}^k_{\infty\omega}$ under arbitrary measures. In *1994 Annual Conference of the European Association for Computer Science Logic - CSL '94*, volume 933 of *Lecture Notes in Computer Science*, pages 228–241. Springer, 1995.

[Mil90] R. Milner. Operational and algebraic semantics of concurrent processes. In J. van Leeuween, editor, *Handbook of Theoretical Computer Science*, volume B, pages 1201–1242. The MIT Press/Elsevier, 1990.

[Mos74] Y. N. Moschovakis. *Elementary Induction on Abstract Structures*. North Holland, 1974.

[Pap94] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley Publishing Company, 1994.

[Pez99] E. Pezzoli. Computational complexity of Ehrenfeucht-fraïssé games on finite structures. In *Proc. 12th International Workshop on Computer Science Logic - CSL '98*, pages 159–170. Springer-Verlag, 1999.

[RS85] N. Robertson and P. D. Seymour. Disjoint paths - a survey. *SIAM Journal of Algebraic and Discrete Methods*, 6:300–305, 1985.

[RS95] N. Robertson and P.D. Seymour. Graph Minors XIII. the disjoint paths problem. *J. Combinatorial Theory B*, 63:65–110, 1995.

[Sch94] T. Schwentick. Graph connectivity and monadic NP. In *Proc. 35th IEEE Symp. on Foundations of Computer Science*, pages 614–622, 1994.

[Sco65] D. Scott. Logic with denumerably long formulas and finite strings of quantifiers. In J.W. Addison, L. Henkin, and A. Tarski, editors, *The Theory of Models*, pages 320–341. North Holland, 1965.

[Spe61] C. Spector. Inductively defind sets of natural numbers. In *Infinitistic Methods*, pages 97–102. Pergamon, 1961.

[SS88] S. Shelah and J. Spencer. Zero-one laws for sparse random graphs. *J. Amer. Math. Soc.*, 1:97–115, 1988.

[Sto77] L. J. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3:1–22, 1977.

[Tar55] A. Tarski. A lattice theoretical fixpoint theorem and its applications. *Pacific J. of Mathematics*, 5:285–309, 1955.

[Ull89] J. D. Ullman. *Database and Knowledge-Base Systems, Volumes I and II*. Computer Science Press, 1989.

[Var82] M. Y. Vardi. The complexity of relational query languages. In *Proc. 14th ACM Symp. on Theory of Computing*, pages 137–146, 1982.

[vB84] J. van Benthem. Correspondence theory. In D.M. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume 2, pages 167–247. Reidel, 1984.