Lectures on Finite Model Theory

Szymon Toruńczyk

Lectures on Finite Model Theory

Contents

1 Overview

Classical model theory mostly focuses on first-order logic, although other logics have also been extensively studied. The main focus there is on *theories*, or sets of sentences. An important goal is to understand the *models* of a given theory, that is, the structures which satisfy all the sentences of the theory. For example, there is a sentence expressing that a given structure equipped with a binary operation $+$ is an abelian group where every element is its own inverse, *i.e.* $x + x = 0$ for all x. The models of this sentence can be easily classified: every model is a vector space over the two-element field and is therefore determined – up to isomorphism – by its dimension, or by its cardinality. Furthermore, for each model, its subsets which are definable by a first-order formula can be analysed and understood – they are Boolean combinations of vector subspaces. Model theory tries to achieve a similar level of understanding not just for specific theories, but for entire classes of theories. A typical question studied in model theory is for example: characterize all theories which have exactly one model – up to isomorphism – in any given infinite cardinality, and understand those models and the definable sets in those models.

In *finite model theory*, one of the aims is to understand the expressive power of various logics on finite structures, especially by comparing those logics with various complexity classes. For example, what is the computational complexity of deciding whether a given first-order sentence φ holds in a given finite structure $\mathbb{A}$? It is not difficult to see that this problem is PSPACE-complete when both φ and $\mathbb{A}$ are given on input. What is the complexity when φ comes from a more expressive logic? Or when φ is fixed? What is it, when $\mathbb{A}$ is assumed to come from a specific class of structures, such as the class of all planar graphs? Can a certain computational problem be defined by a sentence of a considered logic?

Satisfiability and model checking

We will see in Chapter 4 that it is undecidable whether a given first-order sentence is satisfiable in some finite structure, or in any structure. This may change when considering other logics, or other classes of structures. For instance, when restricting to trees, the satisfiability problem becomes decidable.

In this lecture, we will discuss the following topics, whose goal is to understand the expressive power of various logics on finite structures:

- Develop tools for proving that a certain property cannot be expressed in a given logic. One of the main tools will be variants of games, tailored for to the considered logic.

- Is it decidable whether a given sentence is satisfiable in some finite structure? What if we consider finite structures of specific forms, like trees?

- Understand the complexity of evaluating a fixed sentence on a given structure.

- In particular, a formula of a given logic defines a property of finite structures. What is the computational complexity of this property? Is there a polynomial time algorithm deciding the property? Is there a linear time algorithm?

- Conversely, can every polynomial-time property be defined by a formula of a given logic? Find logics which correspond precisely to known complexity classes.

2 Preliminaries

2.1 Structures

A *signature*, or *vocabulary*, or *language* is a set Σ of symbols, where each symbol is either declared a relation symbol or a function symbol and comes with a specified arity, which is a natural number. A *structure* $\mathbb{A}$ over a signature Σ, or a Σ-structure, consists of a domain A together with:

- a relation $R_{\mathbb{A}} \subseteq A^r$ for each relation symbol $R \in \Sigma$ of arity r,
- a function $f_{\mathbb{A}} \colon A^r \to A$ for each function symbol $R \in \Sigma$ of arity r.

$\mathbb{A}$ will be often identified with its domain, so for example we can say that a is an element of $\mathbb{A}$. Also, usually the subscript $\mathbb{A}$ from the relations $R_{\mathbb{A}}$ and functions $f_{\mathbb{A}}$ will be omitted, when $\mathbb{A}$ is clear from the context and no confusion may arise by mistaking the relations and functions for symbols in Σ.

Relations are sometimes called *predicates*. Unary/binary/ternary etc. refers to the arity of a relation or function. A function of arity 0 is a *constant*. Binary relations are sometimes written in infix notation: aRb instead of $(a, b) \in R$ or $R(a, b)$.

Example 2.1.
- a directed graph can be viewed as a structure over the signature consisting of one binary relation $\to$;

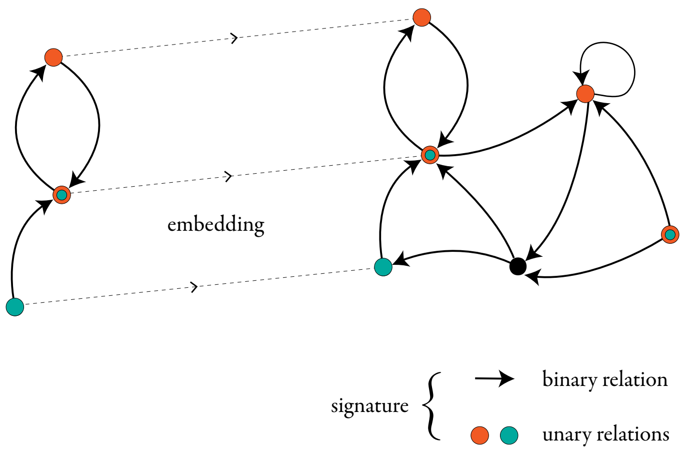


Figure 2.1: Two structures over a signature with two unary relations and one binary relation, and an embedding between them.

- a totally ordered set can be viewed as a structure over the signature consisting of one binary relation $\leqslant$;
- a group can be viewed as a structure over the signature consisting of one binary function $\cdot$;
- a vector space over a field $\mathbb{K}$ can be viewed as a structure with a binary $+$ and unary functions c, for each $c \in \mathbb{K}$, which multiply a given vector by the scalar c. This has an infinite signature if $\mathbb{K}$ is infinite.

Substructures, isomorphisms, embeddings. Let $\mathbb{B}$ be a Σ-structure and let $A \subseteq \mathbb{B}$ be a subset of its domain which is closed under all the functions of $\mathbb{B}$, that is, $f_{\mathbb{B}}(a_1, \ldots, a_k) \in A$ for every function symbol $f \in \Sigma$ of arity k and $a_1, \ldots, a_k \in A$. Then the *substructure* of $\mathbb{B}$ *induced* by A is the structure $\mathbb{A}$ with domain A, together with the relations and functions restricted from $\mathbb{B}$, that is $f_{\mathbb{A}} = f_{\mathbb{B}} \restriction_{A^k}$ and $R_{\mathbb{A}} = R_{\mathbb{B}} \cap A^k$, for every function symbol $f \in \Sigma$ and relation symbol $R \in \Sigma$

of arity k. An *induced substructure* of $\mathbb{B}$ is a substructure $\mathbb{A}$ induced by some set A as above.

An *isomorphism* between two structures $\mathbb{A}$ and $\mathbb{B}$ is a bijection $i\colon \mathbb{A} \to \mathbb{B}$ which preserves all the functions, relations, and their negations. Namely, for all $k \in \mathbb{N}$ and $a_1, \ldots, a_k \in \mathbb{A}$,

- $i(f_{\mathbb{A}}(a_1, \ldots, a_k)) = f_{\mathbb{B}}(i(a_1), \ldots, i(a_k))$ for every function symbol $f \in \Sigma$ of arity k, and
- $(a_1, \ldots, a_k) \in R_{\mathbb{A}} \leftrightarrow (i(a_1), \ldots, i(a_k)) \in R_{\mathbb{B}}$ for every relation symbol $R \in \Sigma$ of arity k.

If we just require that i is injective, rather than bijective, we obtain the notion of an *embedding* of $\mathbb{A}$ into $\mathbb{B}$. Equivalently, an embedding from $\mathbb{A}$ into $\mathbb{B}$ is an injective mapping $e\colon \mathbb{A} \to \mathbb{B}$ such that e is an isomorphism between $\mathbb{A}$ and some induced substructure of $\mathbb{B}$. An *automorphism* of $\mathbb{A}$ is an isomorphism between $\mathbb{A}$ and itself.

2.2 *First-order logic*

Syntax. Fix a signature Σ. Also, fix a countably infinite set of *variables*, denoted x, y, z, etc. A *first-order formula* over Σ is built recursively using the following constructs:

$$\bot, \quad \varphi \vee \psi, \quad \neg\varphi, \quad \exists x.\varphi, \quad R(t_1, \ldots, t_k), \quad t_1 = t_2$$

where φ and ψ are simpler formulas, x is a variable, R is a relation symbol in Σ of arity k and $t_1, \ldots, t_k$ are *terms* built out of function symbols and variables. We use standard syntactic sugar, see Fig. 2.2.

An *atomic formula* is a formula of the form $R(t_1, \ldots, t_k)$ or $t_1 = t_2$, where $t_1, \ldots, t_k$ are terms and R is a relation symbol. A *literal* is an atomic formula or a negation of an atomic formula. A *quantifier-free* formula is a formula without quantifiers, that is, a Boolean combination of atomic formulas. For a formula φ its *free variables* are defined inductively:

notation	*definition*	*meaning*
$\top$	$\neg\bot$	true
$\varphi \wedge \psi$	$\neg(\neg\varphi \vee \neg\psi)$	conjunction
$\exists! x.\varphi(x)$	$\exists x.\varphi(x) \wedge (\forall x'.\varphi(x') \to (x = x'))$	exists exactly one
$\forall x.\varphi$	$\neg\exists x.\neg\varphi$	'for all' quantifier
$\exists\bar{x}$	$\exists x_1 \ldots \exists x_n$ if $\bar{x} = \{x_1, \ldots, x_n\}$	existential quantification of tuples
$\forall\bar{x}$	$\forall x_1 \ldots \forall x_n$ if $\bar{x} = \{x_1, \ldots, x_n\}$	universal quantification of tuples
$x \neq y$	$\neg x = y$	unequal
$\varphi \to \psi$	$\neg\varphi \vee \psi$	implication
$\varphi \leftrightarrow \psi$	$(\varphi \to \psi) \wedge (\psi \to \varphi)$	equivalence
$\bigvee_{i\in I} \varphi_i$	$\varphi_{i_1} \vee \cdots \vee \varphi_{i_r}$ if $I = \{i_1, \ldots, i_r\}$	finite disjunction
$\bigwedge_{i\in I} \varphi_i$	$\varphi_{i_1} \wedge \cdots \wedge \varphi_{i_r}$ if $I = \{i_1, \ldots, i_r\}$	finite conjunction

Figure 2.2: Syntactic sugar

- if φ is atomic its free variables are all the variables appearing in it;
- if φ is a Boolean combination of formulas $\varphi_1, \ldots, \varphi_k$ then the set of free variables of φ is the union of the sets of free variables of φ_i, and
- if $\varphi = \exists x.\psi$ then the free variables of φ are the free variables of ψ without x.

A *sentence* is a formula with no free variables.

Valuations. We denote sets of variables by $\bar{x}, \bar{y}$, etc. Let A be a set and $\bar{x}$ be a set of variables. A *valuation* of $\bar{x}$ in A is a function $\bar{a} \colon \bar{x} \to A$. An *$\bar{x}$-tuple* is a valuation of $\bar{x}$ in some set. The set of all valuations of $\bar{x}$ in A is denoted $A^{\bar{x}}$. If $\bar{x}$ has an implicit enumeration $\bar{x} = \{x_1, \ldots, x_k\}$ then $A^{\bar{x}}$ may be identified with A^k, the k-fold Cartesian product of A. If $\mathbb{A}$ is a structure with domain A then $\mathbb{A}^{\bar{x}}$ is the same as $A^{\bar{x}}$.

If $\bar{x}$ and $\bar{y}$ are two disjoint sets of variables then $\bar{x} \cup \bar{y}$ is written $\bar{x}\bar{y}$, or $\bar{x}y$ in the case when $\bar{y} = \{y\}$ is a singleton. If $\bar{a} \in A^{\bar{x}}$ and $\bar{b} \in A^{\bar{y}}$ for disjoint $\bar{x}$ and $\bar{y}$, then $\bar{a}\bar{b} \in A^{\bar{x}\bar{y}}$ is the unique valuation extending $\bar{a}$ and $\bar{b}$.

term $t(\bar{x})$	semantics $t_{\mathbb{A}} \colon \mathbb{A}^{\bar{x}} \to \mathbb{A}$
y	$\bar{a} \mapsto \bar{a}(y)$
$f(t^1(\bar{x}), \ldots, t^k(\bar{x}))$	$\bar{a} \mapsto f_{\mathbb{A}}(t^1_{\mathbb{A}}(\bar{a}), \ldots, t^k_{\mathbb{A}}(\bar{a}))$

formula $\varphi(\bar{x})$	semantics $\varphi_{\mathbb{A}} \subseteq \mathbb{A}^{\bar{x}}$
$\bot$	$\emptyset$
$\neg\psi(\bar{x})$	$\mathbb{A}^{\bar{x}} - \psi_{\mathbb{A}}$
$\alpha(\bar{x}) \vee \beta(\bar{x})$	$\alpha_{\mathbb{A}} \cup \beta_{\mathbb{A}}$
$\exists y.\psi(\bar{x}y)$	$\{\bar{a}\restriction_{\bar{x}} \mid \bar{a} \in \psi_{\mathbb{A}}\}$
$x = y$	$\{\bar{a} \in \mathbb{A}^{\bar{x}} \mid \bar{a}(x) = \bar{a}(y)\}$
$R(t^1(\bar{x}), \ldots, t^k(\bar{x}))$	$\left\{\bar{a} \in \mathbb{A}^{\bar{x}} \mid (t^1_{\mathbb{A}}(\bar{a}), \ldots, t^k_{\mathbb{A}}(\bar{a})) \in R_{\mathbb{A}}\right\}$

Figure 2.3: Semantics of terms $t(\bar{x})$ and of first-order formulas $\varphi(\bar{x})$. Each term and formula above comes with a distinguished set of variables containing the free variables, which is understood from the context.

If $\bar{a} \in A^{\bar{x}}$ and $a \in A$, then $\bar{a}a$ denotes the valuation extending the valuation $\bar{a} \in A^{\bar{x}x}$ to a variable x not occurring in $\bar{x}$ which maps x to a. For unambiguity, it is convenient to assume that x is the first variable not occurring among $\bar{x}$ with respect to an enumeration of all variables, which is fixed once and for all. We will sometimes denote the tuple $\bar{x}x$ for x as above by $\bar{x} + 1$, and by extension, for $l \in \mathbb{N}$ denote by $\bar{x} + l$ the tuple $\bar{x}$ extended by the first l variables not occurring in $\bar{x}$. In particular, $\bar{x} + (l+1) = (\bar{x} + l) + 1$ and $|\bar{x} + l| = |\bar{x}| + l$.

Semantics. The semantics of a first-order formula is defined by induction on the formula. Fix a signature Σ and a Σ-structure $\mathbb{A}$. If $\bar{x}$ is a set of variables then we may write $\varphi(\bar{x})$ to signify that the free variables of φ are *contained* in $\bar{x}$. Formally, $\varphi(\bar{x})$ is a pair of a formula and a set of variables but, abusing language, we say that $\varphi(\bar{x})$ is a formula. In a given structure $\mathbb{A}$, the formula $\varphi(\bar{x})$ defines a set $\varphi_{\mathbb{A}} \subseteq \mathbb{A}^{\bar{x}}$, defined by induction on $\varphi(\bar{x})$, as in Fig. 2.3.

We say that a tuple $\bar{a} \in \mathbb{A}^{\bar{x}}$ *satisfies* $\varphi(\bar{x})$ in $\mathbb{A}$ if $\bar{a} \in \varphi_{\mathbb{A}}$. This is also denoted

$\mathbb{A}, \bar{a} \models \varphi(\bar{x})$ or $\mathbb{A} \models \varphi(\bar{a})$. If φ is a sentence, then we just say that $\mathbb{A}$ *satisfies* φ, or that $\mathbb{A}$ is a *model* of φ, if $\mathbb{A} \models \varphi$.

Say that two Σ-formulas $\varphi(\bar{x})$ and $\psi(\bar{x})$ are *equivalent* over a class of structures $\mathcal{C}$ if $\varphi_{\mathbb{A}} = \psi_{\mathbb{A}}$ for every Σ-structure $\mathbb{A} \in \mathcal{C}$. When $\mathcal{C}$ is not mentioned, then we implicitly mean the class of all Σ-structures.

Example 2.2. The following sentence expresses that the relation $\leqslant$ is a total order:

$$\begin{aligned} \forall x,y.(x \leqslant y) \vee (y \leqslant x) \quad \wedge \quad & \forall x,y.(x \leqslant y) \wedge (y \leqslant x) \rightarrow (x = y) \quad \wedge \\ & \quad \forall x,y,z.(x \leqslant y) \wedge (y \leqslant z) \rightarrow (x \leqslant z). \end{aligned}$$

When talking about partial orders, we use the syntactic sugar $x < y$ to denote $(x \leqslant y) \wedge (x \neq y)$. The following formula $succ(x,y)$ defines the property that x is a (immediate) successor of y:

$$succ(x,y) \equiv (x > y) \ \wedge \ \neg\exists z.(x < z) \wedge (z < y).$$

The sentence $\forall x.\exists y.\ succ(x,y)$ is equivalent to $\neg\exists x.(x = x)$ on the class of finite total orders, but not on the class of all (finite or infinite) total orders.

Example 2.3. Let Σ consist of a single binary relation E. The following sentence expresses that a Σ-structure is a simple (undirected) graph:

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

In other words, simple graphs are precisely those Σ-structures which satisfy the above sentence.

Example 2.4. Fix $d \in \mathbb{N}$. The following sentence expresses that there is a path between two elements x, y consisting of d edges:

$$\alpha_d \equiv \exists x_1 \exists x_2 \ldots \exists x_{d-1}.E(x,x_1) \wedge \cdots \wedge E(x_{d-1},y)$$

where for $d = 0$ we put $\alpha_0 \equiv (x = y)$ and for $d = 1$ we put $\alpha_1 \equiv E(x,y)$.

Fix a graph G. Let $R_d \subseteq G \times G$ consist of all pairs (a,b) such that the distance from a to b in G is at most d. Then $\varphi_d \equiv \bigvee_{i=0}^{d} \alpha_d(x,y)$ defines the relation R_d in G.

Observe that $R_d(a,b)$ if and only if there some c such that $R_{\lfloor d/2 \rfloor}(a,c)$ and $R_{\lceil d/2 \rceil}(c,b)$. Using this, we may inductively define a formula $\psi_d(x,y)$ with $\log_2(d)$ quantifiers, such that ψ_d defines R_d in G. Hence, ψ_d is equivalent to φ_d on the class of simple undirected graphs.

2.3 Basics of model theory

Elementary equivalence. Say that two structures $\mathbb{A}$ and $\mathbb{B}$ over the same signature Σ are *elementarily equivalent* if they satisfy exactly the same first-order sentences over Σ.

Example 2.5. If $\mathbb{A}$ and $\mathbb{B}$ are isomorphic, then they are elementarily equivalent. Formally, this is proved by showing more generally that if $f\colon \mathbb{A} \to \mathbb{B}$ is an isomorphism and $\varphi(x_1,\ldots,x_k)$ is any first-order formula, then $\mathbb{A} \models \varphi(a_1,\ldots,a_k)$ holds if and only if $\mathbb{B} \models \varphi(f(a_1),\ldots,f(a_k))$, for all $a_1,\ldots,a_k \in \mathbb{A}$. This is proved by a straightforward induction on the structure of the formula φ. Informally, the semantics of first-order logic is invariant under isomorphisms. This also applies to all other reasonable logics.

Two non-isomorphic structures may still be elementarily equivalent.

Example 2.6. The structures $\mathbb{A} = (\mathbb{N},=)$ and $\mathbb{B} = (\mathbb{R},=)$ are elementarily equivalent, but are non-isomorphic. Again, we may prove by induction on the structure of $\varphi(x_1,\ldots,x_k)$ that if $n_1,\ldots,n_k \in \mathbb{N}$ then $\mathbb{A} \models \varphi(n_1\ldots,n_k)$ if and only if $\mathbb{B} \models \varphi(n_1,\ldots,n_k)$, where now $n_1,\ldots,n_k$ are treated as real numbers. In the inductive step, the least trivial case is when $\varphi(x_1,\ldots,x_k)$ is of the form $\exists y.\psi(x_1,\ldots,x_k,y)$ and we want to prove that if $\mathbb{B} \models \varphi(n_1,\ldots,n_k)$ holds then also $\mathbb{A} \models \varphi(n_1,\ldots,n_k)$. By assumption, there is some $r \in \mathbb{R}$ such that $\mathbb{B} \models \psi(n_1,\ldots,n_k,r)$ holds. Now observe that there is an automorphism f of $\mathbb{B} = (\mathbb{R},=)$ that fixes each of the elements $n_1,\ldots,n_k$ and maps r to some natural number m. Indeed, the automorphisms of $\mathbb{B}$ are just the bijections $f\colon \mathbb{R} \to \mathbb{R}$, so we may take f to be the identity if $r \in \mathbb{N}$, and otherwise, we may take f to be the bijection which exchanges r with any number $m \in \mathbb{N} - \{n_1,\ldots,n_k\}$, and fixes all reals numbers other than r and m.

Since f is an automorphism, $\mathbb{B} \models \psi(f(n_1), \ldots, f(n_k), f(r))$ holds, thus $\mathbb{B} \models \psi(n_1, \ldots, n_k, m)$. By inductive assumption, $\mathbb{A} \models \psi(n_1, \ldots, n_k, m)$ and therefore $\mathbb{A} \models \varphi(n_1, \ldots, n_k)$.

Example 2.7. The structures $(\mathbb{N}, \leqslant)$ and $(\mathbb{R}, \leqslant)$ are not elementarily equivalent, as the sentence expressing that there is a least element is satisfied in only one of them. However, by a similar argument as above, one can show that the structures $(\mathbb{R}, \leqslant)$ and $(\mathbb{Q}, \leqslant)$ are elementarily equivalent. The key observation here is that for any fixed rationals $a_1, \ldots, a_k$ and real $r \in \mathbb{R}$ there is an automorphism f of $(\mathbb{R}, \leqslant)$, that is, an increasing bijection, such that f fixes each of the numbers $a_1, \ldots, a_k$, and maps r to a rational number.

Example 2.8. We now show two countable total orders $\mathbb{A}$ and $\mathbb{B}$ that are elementarily equivalent, but non-isomorphic.

Namely, $\mathbb{A} = (\mathbb{Z}, \leqslant)$, whereas $\mathbb{B}$ consists of two copies of $\mathbb{Z}$, one following the other. Formally, the elements of $\mathbb{B}$ are pairs (σ, n) with $\sigma \in \{1, 2\}$ and $n \in \mathbb{Z}$, and those pairs are ordered by $\leqslant$ lexicographically: $(\sigma, m) \leqslant (\tau, n)$ if and only if $\sigma < \tau$ or $\sigma = \tau$ and $m \leqslant n$.

One can prove that $\mathbb{A}$ and $\mathbb{B}$ are elementarily equivalent. This is done using *Ehrenfeucht-Fraïssé games*, which will be developed in Chapter 5 (see Exercise 5.18).

Compactness. Fix a signature Σ. A first-order *theory* is a set of first-order sentences over Σ. Note that if the cardinality of Σ is at most some infinite cardinal κ, then so is the cardinality of T. In other words, T has cardinality at most $\max(|\Sigma|, \aleph_0)$, so if Σ is countable then so is T. A *model* of a theory T is a structure which satisfies all the sentences in the theory. Conversely, if $\mathbb{A}$ is a Σ-structure, then *the theory* of $\mathbb{A}$ is the set of all (first-order) Σ-sentences φ such that $\mathbb{A} \models \varphi$. Thus, two structures $\mathbb{A}$ and $\mathbb{B}$ are elementarily equivalent if and only if their theories are equal.

Note that if T is a theory then it may have models that are not elementarily equivalent.

The most fundamental result of model theory is the *compactness theorem*:

Theorem 2.9 (Compactness of first-order logic). *A first-order theory T has a model if and only if every its finite subset $S \subseteq T$ has a model. In this case, T also has a model of cardinality at most* $\max(\aleph_0, |\Sigma|)$.

The first part in the above statement is the more essential part of the compactness theorem, and this part alone is sometimes called the compactness theorem. It follows easily from *Gödel's completeness theorem*. This theorem states that if a theory T has no model then there is a proof (in some formal system) deriving the contradiction $\perp$ from T. As a proof can only invoke a finite number of sentences from T, this shows that there is some finite subset $S \subseteq T$ from which a contradiction can already be obtained, and, in particular, S has no model. Another proof (without invoking proof systems) is presented in Section 2.4.

The second part in the above statement is the so-called downward Löwenheim-Skolem theorem: if a theory T has a model then it has a model of cardinality at most $\max(\aleph_0, |\Sigma|)$. This part also follows from the proof of Gödel's completeness theorem. Equivalently, the downward Löwenheim-Skolem states that every structure is elementarily equivalent to some structure of cardinality at most $\max(\aleph_0, |\Sigma|)$. In particular, if Σ is a countable signature then every Σ-structure is elementarily equivalent to a countable structure. For instance, the structure $(\mathbb{R}, +, \cdot)$ is elementarily equivalent to a countable structure. Indeed, one can prove that the set $A \subseteq \mathbb{R}$ of algebraic numbers (roots of non-zero polynomials with integer coefficients) is closed under $+$ and $\cdot$, and $(A, +, \cdot)$ is elementarily equivalent to $(\mathbb{R}, +, \cdot)$ (see Exercise 7.8).

Another formulation of the compactness theorem is as follows. For simplicity, we state it for countable signatures Σ only. A sequence of structures $\mathbb{A}_1, \mathbb{A}_2, \ldots$ *converges elementarily* to a structure $\mathbb{A}$ if for every first-order sentence φ, φ holds in $\mathbb{A}$ if and only if φ holds in all but finitely many elements of the sequence $\mathbb{A}_1, \mathbb{A}_2, \ldots$.

Theorem 2.10 (Compactness, restated). *Every infinite sequence of structures over a countable signature contains an infinite subsequence which converges elementarily to some countable structure.*

A proof of Theorem 2.10, without the downward Löwenheim-Skolem theorem, is presented in Section 2.4.

To prove the equivalence of the two statements, the following lemma is useful.

Lemma 2.11. *Let $\mathbb{A}_1, \mathbb{A}_2, \ldots$ be a sequence of structures over a countable signature Σ. Then there is a subsequence $\mathbb{A}_{i_1}, \mathbb{A}_{i_2}, \ldots$ such that for every sentence φ, either $\mathbb{A}_{i_n} \models \varphi$ holds for all but finitely many $n \in \mathbb{N}$, or $\mathbb{A}_{i_n} \models \neg\varphi$ holds for all but finitely many $n \in \mathbb{N}$.*

Exercise 2.12. Prove Lemma 2.11. *Hint: use the fact that there are countably many sentences.*

Exercise 2.13. Prove the equivalence of the two statements of the compactness theorem, for countable signatures.

As a very simple application, we show that graph connectedness cannot be expressed by any theory.

Example 2.14. We show that there is no theory whose models are precisely the connected graphs.

Consider the sequence $P_1, P_2, \ldots$, where P_n is the path with n vertices. By Theorem 2.10, some subsequence converges elementarily to a structure G. Then G is a graph with exactly two vertices of degree 1 and the remaining vertices of degree 2, and has at least n vertices, for all $n \in \mathbb{N}$ (since all these properties hold for almost all the structures in the sequence $P_1, P_2, \ldots$). Moreover, if connectedness were expressible by a set T of first-order sentences, then G would also be a model of T (as all the graphs P_i are connected) and therefore G would be connected. This is a contradiction, since there is no connected graph G with the above properties.

Exercise 2.15. Prove that every theory T that has an infinite model also has a model of cardinality larger than any given cardinal.
Hint: extend the signature by adding as many constant symbols as the size of the cardinal.

Exercise 2.16. Prove that there is a countable structure that is elementarily equivalent to $(\mathbb{Z}, \leqslant)$, but non-isomorphic with it. An explicit example is given in

Example 2.8, but such a structure can be also obtained using compactness.
Hint: extend the signature with two constant symbols c, d, and consider the theory T of $(\mathbb{Z}, \leqslant)$ together with sentences φ_n, for $n \in \mathbb{N}$, where $\varphi_n \equiv \exists x_1, \dots, x_n . c < x_1 < \dots < x_n < d$.

Exercise 2.17. Let T be the first-order theory of $(\mathbb{N}, +, \cdot)$, that is, T is the set of all sentences φ over the signature consisting of two binary function symbols $+$ and $\cdot$ that are true in the structure $(\mathbb{N}, +, \cdot)$. Prove that there exists a countable model of T that is not isomorphic to $(\mathbb{N}, +, \cdot)$.

Exercise 2.18. Prove that for each $n \in \mathbb{N}$ there is a formula $\delta_n(x, y)$ of size $\mathcal{O}(\log n)$ expressing that the distance between x and y in a graph is at most n. The size of a formula φ is the size of its parse tree. In particular, if the same subformula ψ occurs in φ twice, then it contributes twice to its size.

2.4 *Proof of the compactness theorem*[*]

We prove the compactness theorem, in the formulation from Theorem 2.10. We start with defining the notion of an ultrafilter.

An *ultrafilter* on $\mathbb{N}$ is a function $\mu \colon \mathcal{P}(\mathbb{N}) \to \{0, 1\}$ which is a Boolean algebra homomorphism, that is, $\mu(\emptyset) = 0$, $\mu(X \cup Y) = \mu(X) \vee \mu(Y)$ and $\mu(\mathbb{N} - X) = 1 - \mu(X)$. In other words, μ is a $0,1$-valued, finitely additive measure on the set of all subsets of $\mathbb{N}$, that is, $\mu(X \cup Y) = \mu(X) + \mu(Y)$ whenever X and Y are disjoint and $\mu(\mathbb{N}) = 1$.

An example ultrafilter is the ultrafilter μ_k such that $\mu_k(X) = 1$ if and only if $k \in X$, where $k \in \mathbb{N}$ is some fixed number. Ultrafilters of this form are called *principal*. An ultrafilter μ is not principal if and only if $\mu(X) = 0$ for every finite set X (see Exercise 2.22). It follows from the axiom of choice that there exist ultrafilters that are not principal.

Lemma 2.19. *There exists a non-principal ultrafilter on* $\mathbb{N}$.

Proof. See Exercise 2.23. ∎

[*]omitted in the lectures

In the following, fix any non-principal ultrafilter μ. Call a set $X \subseteq \mathbb{N}$ *large* if $\mu(X) = 1$ and *small* otherwise. Note that the union of two small sets is small, and dually, the intersection of two large sets is large.

Let $A_1, A_2, \ldots$ be a sequence of sets. Write $\prod_{n\in\mathbb{N}} A_n$ for their product, that is, the set of all sequences $a_1, a_2, \ldots$ with $a_n \in A_n$ for $n \in \mathbb{N}$. Such a sequence is denoted $\bar{a}_n \in \prod_{n\in\mathbb{N}} A_n$. Now, define an equivalence relation $\sim$ on $\prod_{n\in\mathbb{N}} A_n$, so that $\bar{a}_n \sim \bar{b}_n$ if and only if the set $\{n \in \mathbb{N} \mid a_n = b_n\}$ is large. This is indeed an equivalence relation, where transitivity follows from the fact that the intersection of large sets is large. Write $\prod_\mu A_n$ for the set of $\sim$-equivalence classes; this set is called the *ultraproduct* of the sets $A_1, A_2, \ldots$.

We now proceed to proving Theorem 2.10.

Fix a signature Σ. For simplicity, assume that Σ contains only relation symbols, as functions can be represented as relations (a function $f\colon A^n \to A$ is represented by its graph $\Gamma_f \subseteq A^{n+1}$). Moreover, to simplify the proof, we assume that Σ is countable. The proof in the general case proceeds similarly.

Let $\mathbb{A}_1, \mathbb{A}_2, \ldots$ be a sequence of Σ-structures, and let A_n denote the domain of $\mathbb{A}_n$. Without loss of generality, by using Lemma 2.11 and replacing the sequence $\mathbb{A}_1, \mathbb{A}_2, \ldots$ by its subsequence, we may assume that for every sentence φ, the set $\{n \in \mathbb{N} \mid \mathbb{A}_n \models \varphi\}$ is either finite, or its complement (in $\mathbb{N}$) is finite.

Let $A = \prod_\mu A_n$ be the ultraproduct of the sets $A_1, A_2, \ldots$. Define a new Σ-structure $\mathbb{A}$ with domain A and relations defined as follows. For each realation symbol $R \in \Sigma$ of arity k, let $R_\mathbb{A}$ consist of all k-tuples $([\bar{a}^1_n]_\sim, \ldots, [\bar{a}^k_n]_\sim) \in A^k$ of $\sim$-equivalence classes of sequences $\bar{a}^1_n, \ldots, \bar{a}^k_n \in \prod_{n\in\mathbb{N}} A_n$ such that $R(a^1_n, \ldots, a^k_n)$ holds for all $n \in \mathbb{N}$. The structure $\mathbb{A}$ is called the *ultraproduct* of the structures $\mathbb{A}_1, \mathbb{A}_2, \ldots$.

Lemma 2.20. *For every first-order formula $\varphi(x_1, \ldots, x_k)$ and sequences $\bar{a}^1_n, \ldots, \bar{a}^k_n$ in $\prod_{n\in\mathbb{N}} A_n$, the following conditions are equivalent:*

- $\mathbb{A} \models \varphi([\bar{a}^1_n]_\sim, \ldots, [\bar{a}^k_n]_\sim)$,
- *the set of all $n \in \mathbb{N}$ such that $\mathbb{A} \models \varphi(a^1_n, \ldots, a^k_n)$ is large.*

Proof. By induction on the structure of φ. In the base case, φ is an atomic formula $R(x_1, \ldots, x_k)$, for some $R \in \Sigma$. This case follows by definition of $R_\mathbb{A}$. In the inductive step we consider three cases. The case when φ is of the form $\neg\psi$ follows from the fact that the complement of a large set is small, and vice-versa. The case when φ is of the form $\varphi_1 \vee \varphi_2$ follows from the fact that the union of two sets is large if and only if at least one of them is large. Finally, the case when φ is of the form $\exists y.\psi$ is straightforward. ■

From this we obtain the following.

Corollary 2.21. *Let φ be a sentence. Then $\mathbb{A} \models \varphi$ if and only if the set $\{n \in \mathbb{N} \mid \mathbb{A}_n \models \varphi\}$ is large.*

Recall that for every sentence φ the set $\{n \in \mathbb{N} \mid \mathbb{A}_n \models \varphi\}$ is either finite, or its complement is finite. Moreover, every finite subset of $\mathbb{N}$ is small, and its complement is large. It follows from Corollary 2.21 that $\{n \in \mathbb{N} \mid \mathbb{A}_n \models \varphi\}$ is a complement of a finite set if and only if $\mathbb{A} \models \varphi$. Therefore, the sequence $\mathbb{A}_1, \mathbb{A}_2, \ldots$ converges elementarily to $\mathbb{A}$. This finishes the proof of Theorem 2.10 (without the downward Löwenheim-Skolem theorem).

Exercise 2.22. Prove that an ultrafilter μ is not principal if and only if $\mu(X) = 0$ for every finite set $X \subseteq \mathbb{N}$.

Exercise 2.23. An *ideal* is a family $\mathcal{I} \subseteq \mathcal{P}(\mathbb{N})$ that is downward-closed with respect to inclusion ($X \subseteq Y$ and $Y \in \mathcal{I}$ implies $X \in \mathcal{I}$) and closed under unions ($X, Y \in \mathcal{I}$ implies $X \cup Y \in \mathcal{I}$). A *maximal ideal* is an ideal $\mathcal{I} \neq \mathcal{P}(\mathbb{N})$ such that for every ideal $\mathcal{I}'$ with $\mathcal{I} \subseteq \mathcal{I}'$, either $\mathcal{I} = \mathcal{I}'$ or $\mathcal{I}' = \mathcal{P}(\mathbb{N})$.

1. Using the Kuratowski-Zorn lemma, show that every ideal other than $\mathcal{P}(\mathbb{N})$ is included in a maximal ideal.
2. Prove that if $\mathcal{I}$ is a maximal ideal then $\mu\colon \mathcal{P}(\mathbb{N}) \to \{0, 1\}$ defined as $\mu(X) = 0$ for $X \in \mathcal{I}$ and $\mu(X) = 1$ for $X \notin \mathcal{I}$ is an ultrafilter.
3. Derive Lemma 2.19, by extending the ideal of finite subsets of $\mathbb{N}$ to a maximal ideal.

2.5 Second-order logic

Second-order logic is an extension of first-order logic that allows quantification over sets of elements (denoted $X, Y, \ldots$), and more generally, relations (denoted $R, S, \ldots$) on the domain, rather than just over single elements of the domain (denoted $x, y, \ldots$). The arity d of a relation R is assumed to be fixed, and may be indicated by using a superscript, e.g. $R^{(d)}$.

For example, the formula

$$\varphi(x,y) \equiv \exists X^{(1)}.\ [X(x) \wedge \neg X(y) \wedge \forall z.\forall t.X(z) \wedge E(z,t) \to X(t)]$$

evaluated in a graph G, expresses that x and y are not in the same connected component. The sentence

$$\psi = \exists R^{(2)}.\forall x.\forall y.(R(x,y) \vee R(y,x)) \wedge \forall z.(R(x,y) \wedge R(y,z) \to R(x,z))$$

expresses that there exists a binary relation R which is a total preorder on the domain.

The syntax of second-order logic is obtained by extending that of first-order logic as follows:

- apart from the usual *first-order* variables, denoted $x, y, \ldots$, a countable set of *second-order* variables, denoted $X, Y, \ldots, R, S, \ldots$ is fixed. Each second-order variable has a specified *arity*, which may be indicated in a superscript *e.g.* $R^{(2)}$. If the arity is not indicated, we assume the arity is 1. Relation and function symbols from the signature Σ (as well as $=$) may only be applied to first-order variables;
- a second-order quantifier $\exists X.\varphi$ is added, binding a set variable X;
- for each second-order variable R of arity k and first-order variables $x_1, \ldots, x_k$, $R(x_1, \ldots, x_k)$ is an atomic formula.

A formula may now have free variables which are either first-order variables or second-order variables. A valuation v in a structure $\mathbb{A}$ maps each first-order variable x to an element $v(x)$ of $\mathbb{A}$, while each second-order variable $R^{(k)}$ is mapped to a relation $R_v \subseteq \mathbb{A}^k$ on the domain of $\mathbb{A}$. The semantics of the predicate $R(x_1, \ldots, x_k)$ is defined in the expected way: a valuation v satisfies $R(x_1, \ldots, x_k)$ if $(v(x_1), \ldots, v(x_k)) \in R_v$.

Monadic second-order logic (MSO for short) is the fragment of second-order logic in which only second-order variables of arity 1 are allowed. The formula $\varphi(x, y)$ above is an MSO formula, while the sentence ψ is not.

Exercise 2.24. Show that the compactness theorem fails for monadic second-order logic.

3 Evaluation

Fix a logic $\mathcal{L}$, for example, first-order logic, monadic second-order logic, or second-order logic. The *evaluation*, or *model-checking* problem for $\mathcal{L}$ is the problem of determining whether a given structure $\mathbb{A}$ satisfies a given sentence φ of the logic $\mathcal{L}$. There are three variants of the problem, depending on what is considered fixed:

Combined complexity: Both $\mathbb{A}$ and φ are given on input.

Data complexity: The sentence φ is considered fixed, and only $\mathbb{A}$ is given on input. So this is a family of problems, one for each sentence φ.

Query complexity: The structure $\mathbb{A}$ is considered fixed, and only φ is given on input. So this is a family of problems, one for each structure $\mathbb{A}$.

In each case, we may study the computational complexity of the resulting problem or family of problems. Most of our focus will be on data complexity. In this case, we can for example ask: is the evaluation problem for $\mathcal{L}$ in $\textsc{p}$, for every fixed φ? If this is the case, we may write that $\mathcal{L} \subseteq \textsc{p}$.

In the first and second problem, we need to specify how the structure is represented. In the third problem, as $\mathbb{A}$ is fixed, it is not given on input. In fact, this problem makes sense even for infinite structures $\mathbb{A}$, such as the field of reals $(\mathbb{R}, +, \cdot)$.

3.1 Representing finite structures

For fixed numbers $r, n \in \mathbb{N}$, the *lexicographic order* $\leqslant_{\text{lex}}$ of $\{0, 1, \ldots, n-1\}^r$ is defined as follows. Given two distinct tuples $\bar{a}, \bar{b} \in \{0, 1, \ldots, n-1\}^r$, let $i \in \{1, \ldots, r\}$ be the least such that $\bar{a}[i] \neq \bar{b}[i]$. Then $\bar{a} <_{\text{lex}} \bar{b}$ if $\bar{a}[i] < \bar{b}[i]$.

Fix a finite relational signature $\Sigma = \{R_1, \ldots, R_k\}$. Let $\mathbb{A}$ be a Σ-structure with domain $\{0, \ldots, n-1\}$, for some $n \in \mathbb{N}$. The *adjacency matrix encoding* of $\mathbb{A}$ is the following word over the alphabet $\{0, 1\}$:

$$\underbrace{1 \ldots 1}_{n} 0 [R_1] \ldots [R_k]$$

where for each relation symbol R_i of arity r, $[R_i]$ is the flattening of the r-dimensional $n \times \cdots \times n$ array representing the relation R_i, that is the 0-1 word of length n^r whose jth letter is 1 if and only if the jth tuple in $\{0, \ldots, n-1\}^r$ (with respect to the lexicographic order) belongs to R_i.

As every finite Σ-structure is isomorphic to a structure with domain $\{0, \ldots, n-1\}$, when considering algorithms, we may assume that the inputs are restricted to such structures only. Note that fixing an isomorphism between a finite structure $\mathbb{A}$ and a structure with domain $\{0, \ldots, n-1\}$ amounts to picking a total order on the domain of $\mathbb{A}$, and in principle can be done in $n!$ ways. However, there seems to be no simple way of encoding an arbitrary finite structure by a word without imposing an order of its elements, at least if we require that the encoding can be produced in time polynomial from the encoding just described.

Exercise 3.1. Construct an encoding of finite graphs that is order-invariant, so that two graphs are isomorphic if and only if they have identical encodings. The encoding may take exponential time to produce, given a structure represented in the adjacency matrix encoding as described above.

3.2 First-order logic

The naive algorithm for model checking first-order logic yields the following:

Proposition 3.2. *There is an algorithm which given a Σ-structure $\mathbb{A}$ and a first-order Σ-sentence φ of quantifier rank r decides whether $\mathbb{A}$ satisfies φ in time $\mathcal{O}(|\varphi||\mathbb{A}|^r)$.*

Exercise 3.3. Define the *width* of a formula to be the maximal number of free variables in any of its subformulas. Show that a formula φ of width k can be rewritten in polynomial time into a formula which uses only variables $x_1, \ldots, x_k$ (potentially, reusing variables). Show that Proposition 3.2 can be improved by replacing the exponent r by the width of φ.

The evaluation problem for first-order logic has the following complexity:

- combined complexity: PSPACE-complete,
- query complexity: PSPACE-complete for any $\mathbb{A}$ with $|\mathbb{A}| \geqslant 2$, in P otherwise,
- data complexity: in L, actually, in uniform AC^0. So: FO $\subseteq$ uniform AC^0.

Exercise 3.4. Prove the above.

3.3 *Second-order logic*

The naive algorithm for model checking second-order logic yields the following:

Proposition 3.5. *There is an algorithm which given a Σ-structure $\mathbb{A}$ and a first-order Σ-sentence φ of quantifier rank r decides whether $\mathbb{A}$ satisfies φ in time $\mathcal{O}(|\varphi| \cdot 2^{|\varphi||\mathbb{A}|^k})$, where $k \geqslant 1$ is the maximal arity of a second-order variable in φ.*

The evaluation problem for *monadic* second-order logic (MSO) has the following complexity:

- combined complexity: still PSPACE-complete (note that we assume here that all relations in φ have arity 1).
- query complexity: PSPACE-complete for any $\mathbb{A}$ with $|\mathbb{A}| \geqslant 1$, in P otherwise,
- data complexity: in the polynomial hierachy: SO $\subseteq$ PH. Contains complete problems for each level of the hierarchy, as explained below.

Lemma 3.6. *There is a sentence φ of MSO such that for every SAT instance α one can compute in time polynomial in $|\alpha|$ a structure $\mathbb{A}_\alpha$ such that:*

$$\mathbb{A}_\alpha \models \varphi \qquad \Leftrightarrow \qquad \alpha \text{ is satisfiable.}$$

More generally, fix $k \in \mathbb{N}$. A complete problem for the kth level of the polynomial hierarchy (for the class Σ^p_k) is the problem of deciding validity of a given QBF instance of the form

$$\alpha = \underbrace{\exists^* \forall^* \exists^* \ldots}_{k-1 \text{ alternations}} \beta,$$

where β is a propositional formula.

Lemma 3.7. *There is a sentence of φ_k of MSO such that for every QBF instance α as above one can compute in time polynomial in $|\alpha|$ a structure $\mathbb{A}_\alpha$ such that:*

$$\mathbb{A}_\alpha \models \varphi_k \qquad \Leftrightarrow \qquad \alpha \text{ holds.}$$

Exercise 3.8. Prove the above statements above the evaluation of MSO. When α is a SAT instance, then $\mathbb{A}_\alpha$ can be the directed acyclic graph representing α, whose nodes are the subformulas of α, together with unary predicates marking their types ($\wedge, \vee, \neg$, input gate).

4 Satisfiability

The satisfiability problem is the following decision problem:

> **Problem:** SATISFIABILITY
> **Input:** a sentence φ
> **Decide:** is there some structure $\mathbb{A}$ which satisfies φ?

Actually, there are multiple variants of the problem, depending on the considered logic and class of structures. So for example, we may consider the *finite* satisfiability problem for first-order logic. This means that we require the structure $\mathbb{A}$ to be finite. In the *general* satisfiability problem we allow $\mathbb{A}$ to be an arbitrary structure. Also, we may fix a signature Σ, or let it be arbitrary (given on input together with the formula φ). Finally, instead of first-order logic, we could consider other logics.

In this chapter we will study some instances of the satisfiability problem.

4.1 Trakhtenbrot's theorem

We show that there is a finite signature Σ such that the finite satisfiability problem for first-order logic is undecidable.

First, let us note that for some signatures Σ, the problem is decidable.

Example 4.1. Let Σ be a signature consisting only of unary relation symbols. Then the problem of deciding whether a given Σ-sentence is satisfied in some Σ-structure is decidable.

It follows easily from Theorem 5.8 that if $\mathbb{A}$ and $\mathbb{B}$ are two Σ-structures then $\mathbb{A} \equiv^k \mathbb{B}$ if $\mathbb{A}$ and $\mathbb{B}$ have the same numbers of elements of any given atomic type, counting only up to the threshold k. In this case, an atomic type of an element a is just the set of unary predicates that it satisfies. Hence, up to the equivalence $\equiv^k$, there are only finitely many structures $\mathbb{A}$. Given a sentence φ of quantifier rank k, it is therefore enough to test whether or not φ holds in one of those finitely many structures.

Theorem 4.2. *There is a finite signature Σ consisting of unary and binary relational symbols such that the finite satisfiability problem is undecidable for Σ-formulas.*

Note that by Proposition 3.2, the finite satisfiability problem is recursively enumerable, as for a given sentence φ, it suffices to check whether each finite Σ-structure $\mathbb{A}$ is a model of φ.

Proof. Let $L \subseteq \{0,1\}^*$ be an undecidable problem which is recognized by a Turing machine M. For example, M could be the universal Turing machine, accepting exactly encodings of those Turing machines, which halt on the empty word.

Given a word $w \in \{0,1\}^*$ we show how to effectively construct a sentence φ_w such that φ_w is satisfied in a finite structure if and only if $w \in L$. The signature Σ of φ_w will depend only on M, and will consist of unary and binary relations. Hence, this will prove undecidability of finite satisfiability for Σ-formulas.

The idea is that the sentence φ_w axiomatizes that a given Σ-structure is a colored grid representing an accepting run of M on w.

For two numbers $m, n \in \mathbb{N}$, an $m \times n$ *grid* is the finite structure G with domain $\{1, \ldots, m\} \times \{1, \ldots, n\}$, equipped with two binary relations, $\rightarrow$ and $\uparrow$, as depicted in Fig. 4.1. We would like to write a sentence whose finite models are exactly all the grids. This, however, is impossible, as can be seen by an Ehrenfeucht-Fraïssé argument (see Chapter 5). However, we may achieve a weaker condition, which will turn out to be sufficient.

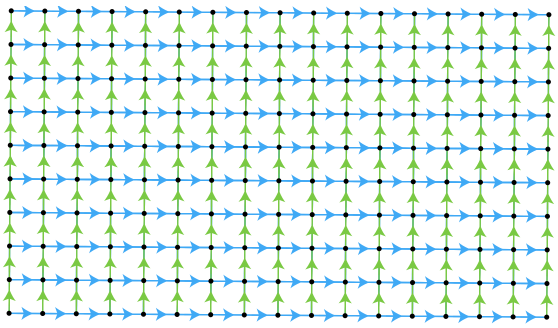


Figure 4.1: A grid. The blue arrows mark the relation $\rightarrow$ and the green arrows mark the relation $\uparrow$.

Let Σ_0 be the signature consisting of the binary relations $\rightarrow$ and $\uparrow$. A grid may be seen as a Σ_0-structure.

Given a Σ_0-structure G, denote by $\leftarrow$ the inverse of $\rightarrow$ and by $\downarrow$ the inverse of $\uparrow$. For a binary relation R, say that b is an R-successor of a if aRb holds.

We define four types of corners in G. For each $H \in \{\leftarrow, \rightarrow\}$ and $V \in \{\uparrow, \downarrow\}$, an *$HV$-corner* is an element which has no H-successor and no V-successor. For each $R \in \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$, an *$R$-boundary* node is an element which has no R-successor.

Let φ_0 be the sentence expressing the following:

- for each $R \in \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$, every element v has at most one R-successor, and this successor is not v;
- for each $H \in \{\leftarrow, \rightarrow\}$ and $V \in \{\uparrow, \downarrow\}$, the relations H and V commute: $H \circ V = V \circ H$;
- there is a unique $\leftarrow\downarrow$-corner, a unique $\leftarrow\uparrow$-corner, a unique $\rightarrow\downarrow$-corner and a unique $\rightarrow\uparrow$-corner.

Clearly, every grid is a model of φ_0. The converse need not hold: consider for example the disjoint union of a grid with a cylinder obtained from another grid by identifying its lower and upper boundaries. However, the connected component of a lower-left corner is always isomorphic to a grid.

The *connected component* of an element v in G is the substructure of G induced by the set of all nodes v' such that $v = v_0 R_0 v_1 R_1 \dots v_{n-1} R_{n-1} v_n = v'$ for some $v_0, \dots, v_n \in G$ with $v_0 = v$, $v_n = v'$ and some $R_0, \dots, R_{n-1} \in \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$.

Lemma 4.3. *Every finite model of φ_0 has a unique $\leftarrow\downarrow$-corner v, and the connected component of v is isomorphic to some grid.*

Proof. Exercise. ∎

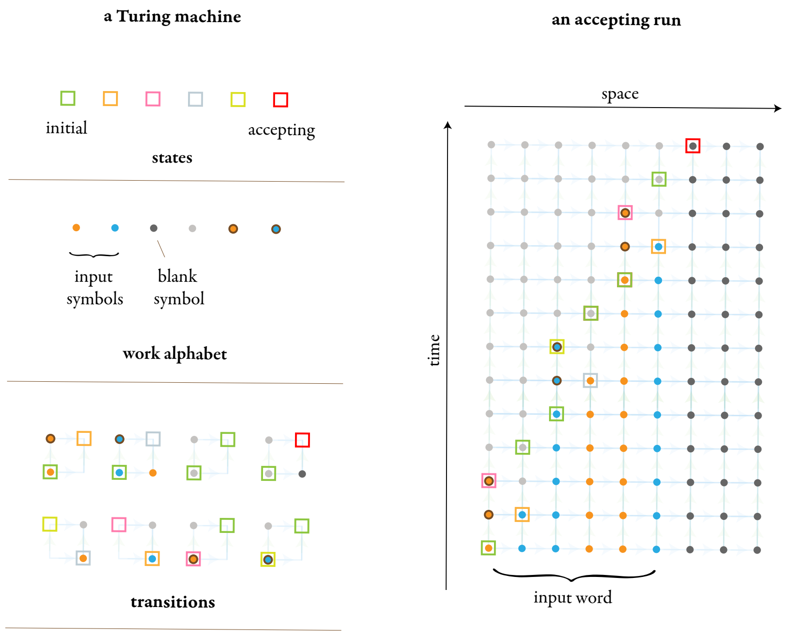


Figure 4.2: A Turing machine and its accepting run.

Finite runs of M may be represented as grids expanded with unary predicates, as shown in Fig. 4.2. There is one unary predicate for each state of the machine and one unary predicate for each symbol of the work alphabet. It is not difficult to write, for each input word w, a sentence ψ_w which holds in a $m \times n$ grid expanded with unary predicates if and only if it is a proper encoding of an accepting run of M on w. It expresses the following properties:

- the first row represents the input word w padded with blank symbols;
- the lower-left node carries the initial state, and no other node does;
- exactly one node carries the accepting state;
- for every node which carries a non-accepting state and its $\uparrow$-successor v, either the $\leftarrow$-successor of v, or the $\rightarrow$-successor of v carries a state;
- for every node which carries a state and is not in the first row, if v is its $\downarrow$-successor, then either the $\leftarrow$-successor of v, or the $\rightarrow$-successor of v carries a state;
- if a node v carries a state which is not accepting then its $\uparrow$-successor v' exists, and either the $\leftarrow$-successor or the $\rightarrow$-sucessor of v' carries a state;
- every 2×2 subgrid is colored consistently with the transition function of M.

In particular, in every model of $\varphi_0 \wedge \psi_w$, the accepting state must occur in the connected component of the lower-left corner v: by tracing the movement of the head upwards, starting from v, we must eventually arrive at an accepting state.

It follows that the sentence $\varphi_0 \wedge \psi_w$ has a finite model if and only if M has an accepting run on w. ■

Exercise 4.4. Prove Lemma 4.3.

Exercise 4.5. Show that for some Turing machine M which does not halt on w, the sentence $\varphi_0 \wedge \psi_w$ constructed in the proof may have an infinite model.

Exercise 4.6. Prove that the *general* satisfiability problem – does a given Σ-sentence have any model – is also undecidable. Using Gödel's completeness theorem, show that the complement of this problem is recursively enumerable.

5
Types

Fix numbers r and $l \in \mathbb{N}$ and a signature Σ. Two l-tuples in a Σ-structure $\mathbb{A}$ have the same *type* of quantifier rank r if they satisfy exactly the same formulas of quantifier rank r. This induces an equivalence relation $\equiv^r$ on l-tuples.

For simplicity, in this section we assume that Σ is a finite and relational signatuer, that is, has no function symbols. We now define types and study their properties.

We first analyze types of quantifier rank 0, and then move to types of higher rank.

5.1 Atomic types

An *atomic type* with variables $\bar{x}$ is a conjunction $\tau(\bar{x})$ of literals such that for every atomic formula $\alpha(\bar{x})$, either α or $\neg\alpha$ appears as a conjunct in τ. Atomic types are also called *quantifier-free types*. Note that a tuple $\bar{a} \in \mathbb{A}^{\bar{x}}$ may satisfy at most one atomic type with variables $\bar{x}$. In fact, it satisfies *exactly* one atomic type. Namely, $\bar{a} \in \mathbb{A}^{\bar{x}}$ induces an atomic type, called *the atomic type* of $\bar{a}$, denoted $\mathrm{atp}_{\mathbb{A}}(\bar{a})$, which is the conjunction of all literals $\alpha(\bar{x})$ or $\neg\alpha(\bar{x})$ which are satisfied by $\bar{a}$ in $\mathbb{A}$.

Here is an example of an atomic type over the signature Σ with one binary relation E, and with variables x, y:

$$(x \neq y) \wedge \neg E(x,x) \wedge E(x,y) \wedge \neg E(y,x) \wedge \neg E(y,y).$$

Above and in the future, we may omit writing some of the conjuncts in an atomic type if they are implied by the remaining conjuncts. For example, we always omit conjuncts $x = x$, and if $x = y$ is a conjunct, then $y = x$ may be omitted. Also, if $x = y$ and $E(x,z)$ are conjuncts, then $E(y,z)$ may be omitted, etc.

An atomic type may be unsatisfiable. For example, over the empty signature, the types $x \neq x$ and $(x = y) \wedge (y = z) \wedge (x \neq z)$ are unsatisfiable, and $(x = y) \wedge E(x,z) \wedge \neg E(y,z)$ cannot be extended to a satisfiable type.

Lemma 5.1. *Let $\mathbb{A}$ and $\mathbb{B}$ be structures. For every two tuples $\bar{a} \in \mathbb{A}^{\bar{x}}$ and $\bar{b} \in \mathbb{B}^{\bar{x}}$, the following conditions are equivalent:*

1. *there is an atomic type $\tau(\bar{x})$ such that $\bar{a} \in \tau_{\mathbb{A}}$ and $\bar{b} \in \tau_{\mathbb{B}}$,*
2. *for every quantifier-free formula $\varphi(\bar{x})$, $\bar{a} \in \varphi_{\mathbb{A}}$ if and only if $\bar{b} \in \varphi_{\mathbb{B}}$,*
3. *the atomic type of $\bar{a}$ is equal to the atomic type of $\bar{b}$, that is, $\mathrm{atp}_{\mathbb{A}}(\bar{a}) = \mathrm{atp}_{\mathbb{B}}(\bar{b})$.*

Proof. Immediate. ∎

Exercise 5.2. Up to equivalence, how many satisfiable types with variables x, y are there over the signature with one binary relation? Give an effective characterization of satisfiable atomic types.

A satisfiable atomic type $\tau(\bar{x})$ can be represented by a Σ-structure $\mathbb{A}_\tau$ whose elements are the parts of the partition of $\bar{x}$ such that two variables x, y are equivalent if and only if $x = y$ is a conjunct of τ (cf. Fig. 5.1). The relations of $\mathbb{A}_\tau$ are defined in the unique way so that $R([x_1], \ldots, [x_k])$ holds in $\mathbb{A}_\tau$, where $[x_i]$ is the part of x_i, if and only if $R(x_1, \ldots, x_k)$ is a conjunct of τ.

Exercise 5.3. Show that the structure $\mathbb{A}_\tau$ above is well-defined and unique.

Exercise 5.4. Show that up to equivalence, there are at most

$$B_l \cdot 2^{|\Sigma| l^m}$$

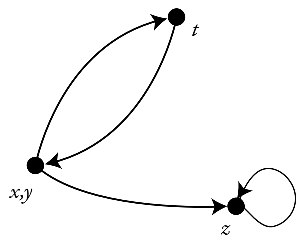


Figure 5.1: An atomic type with variables x, y, z, t over a signature with a binary relation $\to$.

satisfiable atomic types with l variables over a signature Σ consisting of relation symbols of arity at most m. Here, B_l is the lth Bell number, counting the number of distinct partitions of the set $\{1, \ldots, l\}$, and is at most $l!$.

Exercise 5.5. Show that every quantifier-free formula $\varphi(\bar{x})$ is equivalent to a disjunction of satisfiable atomic types, in a unique way. In other words, the satisfiable atomic types form the atoms of the Boolean algebra of quantifier-free formulas with variables $\bar{x}$.

Example 5.6. Let Σ be a relational signature. We characterize when two tuples have equal atomic types. Let $\bar{a} \in \mathbb{A}^l$ and $\bar{b} \in \mathbb{B}^l$ and let A be the set of elements in $\bar{a}$ and B be the set of elements in $\bar{b}$. Then $\text{atp}_{\mathbb{A}}(\bar{a}) = \text{atp}_{\mathbb{B}}(\bar{b})$ if and only if there is a bijection $f\colon A \to B$ such that:

- f maps $\bar{a}$ to $\bar{b}$ componentwise,
- f is an isomorphism from the substructure of $\mathbb{A}$ induced by A to the substructure of $\mathbb{B}$ induced by B. More explicitly, for every relation symbol $R \in \Sigma$ of arity k and tuple $(u_1, \ldots, u_k) \in A^k$, we have $(u_1, \ldots, u_k) \in R_{\mathbb{A}}$ if and only if $(f(u_1), \ldots, f(u_k)) \in R_{\mathbb{B}}$.

5.2 *Types of higher rank*

Given a Σ-structure $\mathbb{A}$ and a tuple of elements $\bar{a} \in \mathbb{A}^{\bar{x}}$, the *type* of quantifier rank r of $\bar{a}$ in $\mathbb{A}$ is the set of all formulas $\varphi(\bar{x})$ of quantifier rank r such that $\bar{a} \in \varphi_{\mathbb{A}}$.

Example 5.7. Two tuples $\bar{a} \in \mathbb{A}^{\bar{x}}$ and $\bar{b} \in \mathbb{B}^{\bar{x}}$ have equal types of quantifier rank 0 if and only if they have equal atomic types, by definition.

As in general, types are infinite objects, to understand them, it is useful to study the equivalence relation of having the same type. It can be characterized in terms of games, as follows.

Fix r and l. Let $\mathbb{A}, \mathbb{B}$ be structures and $\bar{a} = (a_1, \ldots, a_l) \in \mathbb{A}^l$ and $\bar{b} = (b_1, \ldots, b_l) \in \mathbb{B}^l$. We consider a game played on $(\mathbb{A}, \bar{a})$ and $(\mathbb{B}, \bar{b})$ between two players, *spoiler* and *duplicator*. Each of the players has a set of $l + r$ *pebbles*, which will be placed on the elements of $\mathbb{A}$ and $\mathbb{B}$ (cf. Fig. 5.2).

Initially, the first l pebbles of spoiler are placed on the elements $a_1, \ldots, a_l$, and the first l pebbles of duplicator are placed on the elements $b_1, \ldots, b_l$.

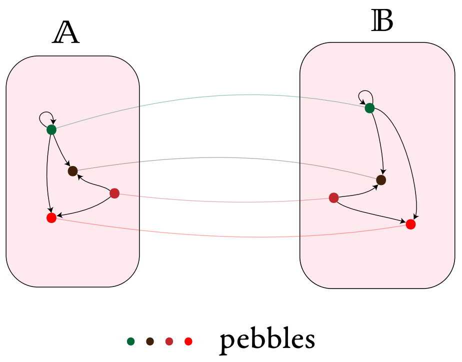


Figure 5.2: The Ehrenfeucht-Fraïssé game. Spoiler places a pebble on one of the structures $\mathbb{A}$ or $\mathbb{B}$ and duplicator places a corresponding pebble on the other structure. The correspondence between the pebbles on $\mathbb{A}$ and $\mathbb{B}$ must be a partial isomorphism in order for duplicator not to lose.

The game then proceeds in r rounds, where in each round, spoiler places a pebble on some element in $\mathbb{A}$ or $\mathbb{B}$, and duplicator responds by placing a pebble on some element of the other structure.

After r rounds, if the substructures of $\mathbb{A}$ and $\mathbb{B}$ induced by the $l+r$ pebbles are identical, then duplicator wins, and loses otherwise. More precisely, for $i = 1, \ldots, r$, let a_{l+i} be the element of $\mathbb{A}$ that was pebbled in the ith round, and b_{l+i} be the element of $\mathbb{B}$ that was pebbled in the ith round. Duplicator wins if

$$\mathrm{atp}_{\mathbb{A}}(a_1, \ldots, a_{l+r}) = \mathrm{atp}_{\mathbb{B}}(b_1, \ldots, b_{l+r}).$$

For two tuples $\bar{a} \in \mathbb{A}^l$ and $\bar{b} \in \mathbb{B}^l$ in structures $\mathbb{A}$ and $\mathbb{B}$, respectively, write $\bar{a} \simeq^r \bar{b}$ if duplicator has a winning strategy in the r-round game on $(\mathbb{A}, \bar{a})$ and $(\mathbb{B}, \bar{b})$.

Instead of defining the notion of a winning strategy precisely, it is easier to formally define the equivalence $\simeq^r$ by induction on r, as follows. Let $\bar{a} \in \mathbb{A}^{\bar{x}}$ and $\bar{b} \in \mathbb{B}^{\bar{x}}$ be tuples of elements of structures $\mathbb{A}$ and $\mathbb{B}$, respectively. Then:

- $\bar{a} \simeq^0 \bar{b}$ if $\mathrm{atp}_{\mathbb{A}}(\bar{a}) = \mathrm{atp}_{\mathbb{B}}(\bar{b})$
- for $r > 0$, $\bar{a} \simeq^r \bar{b}$ if for every $a \in \mathbb{A}$ there is some $b \in \mathbb{B}$ with $\bar{a}a \simeq^{r-1} \bar{b}b$, and conversely, for every $b \in \mathbb{B}$ there is some $a \in \mathbb{A}$ with $\bar{a}a \simeq^{r-1} \bar{b}b$.

Theorem 5.8 (Ehrenfeucht-Fraïssé Theorem). *Fix a relational signature Σ and a number $r \in \mathbb{N}$. Let $\bar{a} \in \mathbb{A}^{\bar{x}}$ and $\bar{b} \in \mathbb{B}^{\bar{x}}$ be tuples of elements of structures $\mathbb{A}$ and $\mathbb{B}$, respectively. Then $\bar{a} \equiv^r \bar{b}$ if and only if $\bar{a} \simeq^r \bar{b}$.*

Example 5.9. Let G be a path and H be a cycle. The following sentence, expressing that there is a node of degree 1, distinguishes G from H:

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

Correspondingly, spoiler wins in the three-round pebble game between G and H: in the first round, he places a pebble on one of the vertices of G of degree 1; duplicator replies by placing his pebble on some vertex b of H. In the next two rounds, spoiler places two pebbles on the two neigbhours of b in H, and duplicator is doomed.

It may be illuminating to check that spoiler has a winning strategy in the r-round game on any path with at most $2^r - 2$ vertices and any other path.

Proof of the Ehrenfeucht-Fraïssé theorem

Let $\bar{x}$ be a finite set of variables and $r \in \mathbb{N}$.

Representing games by trees. Fix a number $r \in \mathbb{N}$, corresponding to the quantifier rank. With each tuple $\bar{a} \in \mathbb{A}^{\bar{x}}$ in a structure $\mathbb{A}$ we associate a finite tree which represents all possible moves of either of the players in the Ehrenfeucht-Fraïssé game, starting from the structure $\mathbb{A}$ with the tuple $\bar{a}$. In the Ehrenfeucht-Fraïssé game on $(\mathbb{A}, \bar{a})$ and $(\mathbb{B}, \bar{b})$ in each of the r rounds, one of the two players chooses to extend $\bar{a}$ in $\mathbb{A}$ by a single element, while the other player extends the tuple $\bar{b}$ in $\mathbb{B}$, and the game continues with the extended tuples. Consider the tree $T^r(\bar{a})$ of possible extensions of $\bar{a}$ in $\mathbb{A}$ for r rounds (cf. Fig. 5.3). It has depth r, and a node at depth i is a tuple extending $\bar{a}$ by i elements of $\mathbb{A}$. The leaves are tuples in $\mathbb{A}^{\bar{x}+r}$, and are labelled by their atomic types.

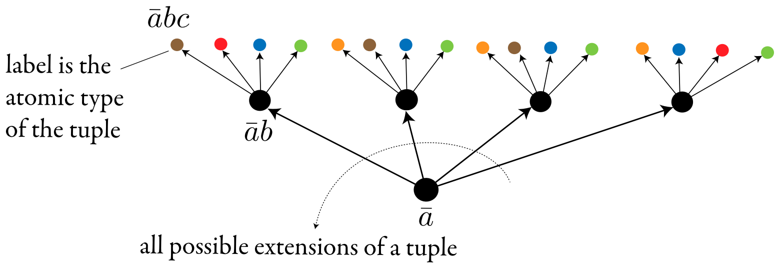

$\bar{a}bc$
label is the atomic type of the tuple
$\bar{a}b$
$\bar{a}$
all possible extensions of a tuple

Figure 5.3: Tree of possible extensions of a tuple $\bar{a}$ by up to 2 elements.

The Ehrenfeucht-Fraïssé game on the pair $(\mathbb{A}, \bar{a})$ and $(\mathbb{B}, \bar{b})$ may be viewed as a game on the trees $T^r(\bar{a})$ and $T^r(\bar{b})$. The game starts with a pebble being placed on the root of $T^r(\bar{a})$ and another pebble being placed on the root of $T^r(\bar{b})$. In each round, spoiler moves the pebble in one of the trees along an edge (*i.e.*, to one of the children of the current node), and duplicator moves the pebble in the other tree along an edge. Once the two pebbles arrive at leaves, duplicator wins if their labels are equal.

The tree $T^r(\bar{a})$ has size $|\mathbb{A}|^r$ which is potentially unbounded in terms of $|\mathbb{A}|$, or even infinite if $\mathbb{A}$ is infinite. However, if two sibling subtrees are isomorphic (as labeled trees), then removing one of them does not affect the game. Hence, we may prune the tree, by removing all isomorphic sibling trees. This amounts to looking at a tree as a set of subtrees, rather than a multiset of subtrees.

This leads to the following, modified definition of a tree, denoted $\mathrm{tree}^r_{\mathbb{A}}(\bar{a})$:

- $\mathrm{tree}^0_{\mathbb{A}}(\bar{a})$ is the atomic type $\mathrm{atp}_{\mathbb{A}}(\bar{a})$ of $\bar{a} \in \mathbb{A}^{\bar{x}}$;
- $\mathrm{tree}^r_{\mathbb{A}}(\bar{a})$, for $r > 0$, is $\left\{\mathrm{tree}^{r-1}_{\mathbb{A}}(\bar{a}a) \mid a \in \mathbb{A}\right\}$, the set (with no duplicates) of all trees of depth $r-1$ for all possible extensions of $\bar{a}$ by one node.

Hence, $\mathrm{tree}^r_{\mathbb{A}}(\bar{a}) \in \mathcal{P}^r(\Delta(\bar{x}+r))$, where $\Delta(\bar{x}+r)$ is the set of satisfiable atomic types of $(\bar{x}+r)$-tuples of elements over the signature Σ and $\mathcal{P}^r$ is the r-fold powerset: $\mathcal{P}^0(\Delta) = \Delta$ and $\mathcal{P}^{r+1}(\Delta) = \mathcal{P}(\mathcal{P}^r(\Delta))$. In particular, the number of such trees can be bounded in terms of r only (independently of $|\mathbb{A}|$) by

$$\left.2^{2^{2^{\cdot^{\cdot^{\cdot^{2^{k}}}}}}}\right\}{r} \tag{5.1}$$

where $k = |\Delta(\bar{x}+r)|$ can be bounded as in Exercise 5.4.

It is not difficult to see that $\mathrm{tree}^r_{\mathbb{A}}(\bar{a}) = \mathrm{tree}^r_{\mathbb{B}}(\bar{b})$ if and only if $\bar{a} \simeq^r \bar{b}$. Indeed, suppose $\mathrm{tree}^r_{\mathbb{A}}(\bar{a}) = \mathrm{tree}^r_{\mathbb{B}}(\bar{b})$. Then in the game played on the (pruned) trees, duplicator can win, by maintaining the invariant that the subtrees induced by the two pebbled nodes are identical. Conversely, if the two trees are distinct, then spoiler can always move one of the pebbles in the direction of a subtree which occurs only in one of the two trees, and will eventually force duplicator to place his pebble on a leaf with the wrong label. This yields:

Lemma 5.10. *Fix $r \in \mathbb{N}$ and let $\bar{a} \in \mathbb{A}^{\bar{x}}$ and $\bar{b} \in \mathbb{B}^{\bar{x}}$ be two tuples of elements in structures $\mathbb{A}$ and $\mathbb{B}$, respectively. Then $\text{tree}^r_{\mathbb{A}}(\bar{a}) = \text{tree}^r_{\mathbb{B}}(\bar{b})$ if and only if $\bar{a} \simeq^r \bar{b}$.*

Proof. We turn the argument above into a more formal proof. Both implications are similar, so we only show the left-to-right implication.

The proof is by induction on r. The base case is immediate. Pick $a \in \mathbb{A}$ (spoiler's move). As $\text{tree}^r_{\mathbb{A}}(\bar{a}) = \text{tree}^r_{\mathbb{B}}(\bar{b})$, there is $b \in \mathbb{B}$ (duplicator's response) such that $\text{tree}^{r-1}_{\mathbb{A}}(\bar{a}a) = \text{tree}^{r-1}_{\mathbb{B}}(\bar{b}b)$. By inductive assumption, $\bar{a}a \simeq^{r-1} \bar{b}b$ (duplicator wins). Symmetrically, for each $b \in \mathbb{B}$ there is $a \in \mathbb{A}$ such that $\bar{a}a \simeq^{r-1} \bar{b}b$. This proves $\bar{a} \simeq^r \bar{b}$. ■

The fact that a given tuple $\bar{a} \in \mathbb{A}^{\bar{x}}$ defines a specific tree t can be defined using a formula, as follows.

Lemma 5.11. *Fix $r \in \mathbb{N}$ and a set of variables $\bar{x}$. For every tree $t \in \mathcal{P}^r(\Delta(\bar{x} + r))$ there is a formula $\widehat{t}(\bar{x})$ of quantifier rank r such that for every structure $\mathbb{A}$ and tuple $\bar{a} \in \mathbb{A}^{\bar{x}}$, the tuple $\bar{a}$ satisfies $\widehat{t}$ in $\mathbb{A}$ if and only if $\text{tree}^r_{\mathbb{A}}(\bar{a}) = t$.*

Proof. By induction on r. The base case is trivial. In the inductive step, write

$$\widehat{t}(\bar{x}) = \bigwedge_{s \in t} \exists y. \widehat{s}(\bar{x}y) \wedge \bigwedge_{s \notin t} \neg \exists y. \widehat{s}(\bar{x}y),$$

where y is a variable not occurring in $\bar{x}$, $\widehat{s}(\bar{x}y)$ is obtained by induction, and in the second conjunction, s ranges over all trees in $\mathcal{P}^r(\Delta(\bar{x} + r)) - t$. It is easy to check that $\widehat{t}(\bar{x})$ satisfies the required condition. ■

Proof of the Ehrenfeucht-Fraïssé Theorem. Theorem 5.8 is implied by the following.

Proposition 5.12. *Fix $r \in \mathbb{N}$ and let $\bar{a} \in \mathbb{A}^{\bar{x}}$ and $\bar{b} \in \mathbb{B}^{\bar{x}}$ be two tuples of elements of two structures $\mathbb{A}$ and $\mathbb{B}$, respectively. The following conditions are equivalent:*

1. $\bar{a} \equiv^r \bar{b}$,
2. $\text{tree}^r_{\mathbb{A}}(\bar{a}) = \text{tree}^r_{\mathbb{B}}(\bar{b})$,
3. $\bar{a} \simeq^r \bar{b}$.

Proof. (1)→(2) follows by Lemma 5.11, since if $t = \mathrm{tree}^r_{\mathbb{A}}(\bar{a})$ then $\bar{a}$ satisfies $\widehat{t}$ and so $\bar{b}$ satisfies $\widehat{t}$ by assumption, proving that $t = \mathrm{tree}^r_{\mathbb{B}}(\bar{b})$.

(2)→(3) is by Lemma 5.10.

(3)→(1). Assume $\bar{a} \simeq^r \bar{b}$. We show that no formula $\varphi(\bar{x})$ of quantifier rank r distinguishes $\bar{a}$ and $\bar{b}$, that is, $\bar{a} \in \varphi_{\mathbb{A}}$ if and only if $\bar{b} \in \varphi_{\mathbb{B}}$. The proof is by induction on r. The base case is immediate, so assume $r > 0$ and that the statement holds for $r - 1$.

Recall that φ is a Boolean combination of formulas of the form $\exists y.\psi(\bar{x}y)$, where ψ has quantifier rank $r - 1$. It is enough to consider the case when φ is of the form $\exists y.\psi$, since if no such formula can distinguish $\bar{a}$ from $\bar{b}$ then neither can a Boolean combination of such formulas.

Suppose $\bar{a} \in \varphi_{\mathbb{A}}$; we show that $\bar{b} \in \varphi_{\mathbb{B}}$. As $\varphi = \exists y.\psi$, there is $a \in \mathbb{A}$ such that $\bar{a}a \in \psi_{\mathbb{A}}$. Since $\bar{a} \simeq^r \bar{b}$, there is some $b \in \mathbb{B}$ such that $\bar{a}a \simeq^{r-1} \bar{b}b$. By inductive assumption, $\psi(\bar{x}y)$ cannot distinguish $\bar{a}a$ and $\bar{b}b$. Hence, $\bar{b}b \in \psi_{\mathbb{B}}$, proving $\bar{b} \in \varphi_{\mathbb{B}}$, as required. ■

As there are finitely many trees in $\mathcal{P}^r(\Delta(\bar{x} + r))$, we get:

Corollary 5.13. *Fix $r \in \mathbb{N}$ and $\bar{x}$. There are finitely many types of quantifier rank r of $\bar{x}$-tuples.*

By the equivalence (1)↔(2) above, defining the same tree of rank r is equivalent to having the same type of rank r. It follows that if $\bar{a} \in \mathbb{A}^{\bar{x}}$ is a tuple, $t = \mathrm{tree}^r_{\mathbb{A}}(\bar{a})$ is its tree and $\widehat{t}(\bar{x})$ is the formula from Lemma 5.11, then $\widehat{t}(\bar{x})$ determines $\mathrm{tp}^r_{\mathbb{A}}(\bar{a})$, meaning that for every $\varphi(\bar{x})$ of quantifier rank r such that $\bar{a}$ satisfies φ in $\mathbb{A}$, the implication $\widehat{t}(\bar{x}) \to \varphi(\bar{x})$ is a tautology. Put in another way, for any tuple $\bar{b} \in \mathbb{B}^{\bar{x}}$ satisfying $\widehat{t}(\bar{x})$, the equivalence $\bar{a} \equiv^r \bar{b}$ holds.

Corollary 5.14. *For every tuple $\bar{a} \in \mathbb{A}^{\bar{x}}$ there is a formula $\tau(\bar{x})$ of quantifier rank r which determines $\mathrm{tp}^r_{\mathbb{A}}(\bar{a})$.*

The above corollary allows to represent each type of quantifier rank r with variables $\bar{x}$ by a single formula $\tau(\bar{x})$ which determines $\mathrm{tp}^r_{\mathbb{A}}(\bar{a})$. By abuse of language, call any such formula $\tau(\bar{x})$ *the type of $\bar{a}$ of quantifier rank r*. Any such formula is equivalent to $\hat{t}(\bar{x})$, for a unique $t \in \mathcal{P}^r$. In particular:

Corollary 5.15. *Every formula $\varphi(\bar{x})$ of quantifier rank r is equivalent to a disjunction of types $\tau(\bar{x})$ of quantifier rank r.*

Proof. For each structure $\mathbb{A}$ and each tuple $\bar{a}$ which satisfies φ in $\mathbb{A}$, let $\tau_{\bar{a}}(\bar{x})$ be the type of $\bar{a}$ of quantifier rank r. Up to equivalence, there are only finitely many formulas of the form $\tau_{\bar{a}}(\bar{x})$, as there are only finitely many types. Then $\varphi(\bar{x})$ is equivalent to the disjunction of all such formulas. ■

More generally, we have the following:

Corollary 5.16. *Let P be a property of pairs $(\mathbb{A}, \bar{a})$, where $\mathbb{A}$ is a Σ-structure and $\bar{a} \in \mathbb{A}^{\bar{x}}$, such that P only depends on the type of $\bar{a}$ of quantifier rank r. Then P is definable by some first-order formula $\varphi(\bar{x})$ of quantifier rank r, so that $(\mathbb{A}, \bar{a})$ satisfies P if and only if $\bar{a} \in \varphi_{\mathbb{A}}$.*

Proof. Same as in Corollary 5.15. ■

The proof of Corollary 5.15 is non-constructive. We give an alternative, constructive proof below.

Lemma 5.17. *There is an algorithm which inputs a formula and outputs an equivalent formula which is a disjunction of formulas of the same quantifier rank which are either types or are unsatisfiable.*

Proof. By induction on r we show that any given formula $\varphi(\bar{x})$ of quantifier rank r is effectively equivalent to a disjunction of formulas of the form $\widehat{t}(\bar{x})$, for $t \in \mathcal{P}^r(\Delta(\bar{x} + r))$.

The base case is Exercise 5.5. Suppose the statement holds for $r - 1$, we show show it for r.

The formula $\varphi(\bar{x})$ is a Boolean combination of formulas of the form $\exists y. \psi(\bar{x}y)$, where ψ has quantifier rank $r - 1$. By inductive assumption, we may assume that ψ is a disjunction of formulas $\widehat{t}(\bar{x}y)$ for $t \in \mathcal{P}^{r-1}(\Delta(\bar{x} + r))$. As disjunctions commute with existential quantification, $\varphi(\bar{x})$ is effectively equivalent to a Boolean combination of formulas of the form $\exists y. \widehat{t}(\bar{x}y)$ as above.

We now show that φ is effectively equivalent to a formula

$$\psi = \psi_1 \vee \ldots \vee \psi_k$$

where each disjunct is of the form

$$\psi_i(\bar{x}) = \bigwedge_{t \in \mathcal{P}^{r-1}(\Delta(\bar{x}+r))} \pm \exists y.\widehat{t}(\bar{x}y),$$

where $\pm$ means the conjunct $\exists y.\widehat{t}$ may be negated. Note that the formula ψ_i above is equal to $\widehat{s}(\bar{x})$, where $s \in \mathcal{P}^r(\Delta(\bar{x}+r))$ is the set of those trees $t \in \mathcal{P}^{r-1}(\Delta(\bar{x}+r))$ which occur in ψ_i positively. Hence, ψ is of the required form.

It therefore remains to convert any Boolean combination $\varphi(\bar{x})$ of formulas of the form $\exists y.\widehat{t}$ into a disjunction ψ as above. This can be reduced to the case of atomic formulas, as follows. For each tree $t \in \mathcal{P}^{r-1}(\Delta(\bar{x}+1))$ introduce a new relation symbol R_t of arity $|\bar{x}|$, and denote the signature consisting of all those symbols by Σ'.

Suppose $\bar{x} = \{x_1, \ldots, x_l\}$. Replace each formula $\exists y.\widehat{t}(\bar{x}y)$ occurring in $\varphi(\bar{x})$ by the atom $R_t(x_1, \ldots, x_l)$. This yields a quantifier-free formula $\varphi'(\bar{x})$ over the signature Σ'. By the quantifier-free case, $\varphi'(\bar{x})$ is equivalent to a formula $\psi'(\bar{x})$ which is a disjunction of atomic types over Σ'. Rewrite ψ' into a Σ-formula by replacing each atom $R_t(\bar{x})$ back by $\exists y.\widehat{t}(\bar{x}y)$, yielding a formula ψ of the required form. ■

Exercise 5.18. Show that the two structures from Example 2.8 are elementarily equivalent.

Exercise 5.19. Let T and T' be two countable trees (connected acyclic graphs) defined as follows. The tree T has a root $\perp$ to which one path of each finite length $n \in \mathbb{N}$ is attached. Formally, T has vertices $\perp$ and (n, i) for $i, n \in \mathbb{N}$ with $i \leqslant n$, and edges $\perp$-$(0, n)$ and (i, n)-$(i+1, n)$ for $i, n \in \mathbb{N}$ with $i < n$.

The tree T' is obtained from T by additionally attaching one infinite path to the root $\perp$. Formally, T' has the same vertices and edges as T, and additionally has vertices (ω, i) for $i \in \mathbb{N}$, and edges $\perp$-$(0, \omega)$ and (i, ω)-$(i+1, \omega)$ for $i \in \mathbb{N}$.

Prove that T and T' are elementarily equivalent.

Exercise 5.20. Show that there is a finite relational signature Σ for which it is undecidable, given $r \in \mathbb{N}$ and $t \in \mathcal{P}^r(\Delta(\bar{x}+r))$, whether $\widehat{t}(\bar{x})$ is satisfiable. Conclude that Lemma 5.17 cannot be improved by removing the unsatisfiable disjuncts from the output.

Exercise 5.21. Fix $r \in \mathbb{N}$ and a set of variables $\bar{x}$. Then formulas $\varphi(\bar{x})$ of quantifier rank r, up to equivalence, form a *boolean algebra* under $\vee, \wedge, \neg$, which we denote $\mathbf{B}^r(\bar{x})$.

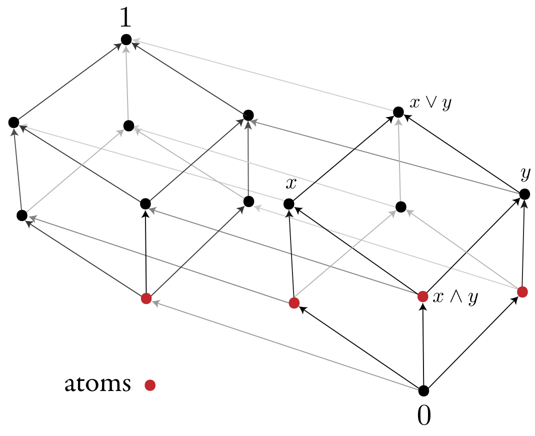


Figure 5.4: A Boolean algebra with 2^4 elements and 4 atoms. Only the Hasse diagram of $\to$ is presented, that is, a set of edges whose transitive closure is $\to$.

Recall that a Boolean algebra $\mathbf{B}$ is a set equipped with operations $\vee, \wedge, \neg$ and constants 0 and 1, satisfying the well-known identities, such as commutativity of $\vee$ and $\wedge$, distributivity of $\vee$ over $\wedge$, $x \vee 1 = 1$ and $x \wedge 0 = 0$, and de Morgan's law $\neg(x \vee y) = \neg x \wedge \neg y$. Every finite Boolean algebra is isomorphic to a Boolean algebra of the form $(\mathcal{P}(X), \cup, \cap, (-)^c, \emptyset, X)$, denoted $\mathcal{P}(X)$ below.

Every Boolean algebra comes with a partial order $\to$, where $x \to y$ iff $\neg x \vee y = 1$ (cf. Fig. 5.4). Nonzero elements of $\mathbf{B}$ which are minimal under $\to$ are called *atoms* of $\mathbf{B}$. As an example, in $\mathcal{P}(X)$, the order $\to$ is just $\subseteq$, and the atoms are the singletons $\{x\}$ for $x \in X$.

In a finite Boolean algebra $\mathbf{B}$, every element $x \in \mathbf{B}$ is equal to the disjunction of all the atoms y with $y \to x$. From this it follows that $\mathbf{B}$ is isomorphic to $\mathcal{P}(\text{Atoms}(\mathbf{B}))$, via the mapping which maps an element $x \in \mathbf{B}$ to the set of atoms y with $y \to x$.

Show that the atoms of the Boolean algebra $\mathbf{B}^r(\bar{x})$ are exactly all types $\tau(\bar{x})$ of quantifier rank r, or, equivalently, all satisfiable formulas $\widehat{t}(\bar{x})$ for $t \in \mathcal{P}^r(\Delta(\bar{x}+r))$. In particular, every formula $\varphi(\bar{x})$ of quantifier rank r is equivalent to a unique disjunction of types $\tau(\bar{x})$.

Exercise 5.22. Design a variant of the Ehrenfeucht-Fraïssé game for MSO, and prove an analogue of Theorem 5.8 for MSO formulas of quantifier rank q. Give an upper bound on the number of types of quantifier rank q corresponding to (5.1).

6
Pebble games

We have seen that Ehrenfeucht-Fraïssé games can be used to determine whether or not two tuples satisfy exactly the same formulas of a given quantifier rank. First-order formulas can be also stratified by other natural parameters, such as number of variables. Those will be useful in further chapters. We will describe games characterizing when two tuples satisfy the same formulas for which those parameters are bounded.

6.1 *Ehrenfeucht-Fraïssé games of infinite duration*

In the previous section, we defined Ehrenfeucht-Fraïssé q-round games in such a way that we only determine the winner in the last round of the game: if the two obtained tuples have equal atomic types, then Duplicator wins, otherwise, Spoiler wins. We could modify the game slightly so that the equality of the atomic types is verified after each move of Duplicator. This would not affect the winner of the game, since if after the ith round the obtained tuples do not have equal atomic types, then the same will hold after the qth round, for $q \geqslant i$, no matter how Duplicator plays.

The advantage of this reformulation of the game is that now we can also play the game for an infinite number of rounds. In fact, we could also consider games whose number of rounds is specified by any ordinal number α, but for simplicity, let us consider games with ω-rounds only. Thus, in this game on $\mathbb{A}$ and $\mathbb{B}$, there are rounds $0, 1, 2, \ldots$, and in the ith round we have elements $a_1, \ldots, a_i$ in $\mathbb{A}$ and $b_1, \ldots, b_i$ in $\mathbb{B}$, and Spoiler extends one of those tuples by an element a_{i+1} or b_{i+1}, respectively, and Duplicator extends the other tuple by the other element b_{i+1} or a_{i+1}. Duplicator loses if $\mathrm{atp}_{\mathbb{A}}(a_1, \ldots, a_{i+1}) \neq \mathrm{atp}_{\mathbb{B}}(b_1, \ldots, b_{i+1})$, and otherwise, the game procedes to round $i+1$. Duplicator wins if she does not lose in any of the rounds $0, 1, 2, \ldots$.

Given two structures $\mathbb{A}$ and $\mathbb{B}$ we may now consider two conditions:

1. for every $q \in \mathbb{N}$, Duplicator wins the q-round Ehrenfeucht-Fraïssé game on $\mathbb{A}$ and $\mathbb{B}$,

2. Duplicator wins the ω-round Ehrenfeucht-Fraïssé game on $\mathbb{A}$ and $\mathbb{B}$.

Clearly, the second condition implies the first one. What about the reverse implication?

Note that the first condition is equivalent to stating that $\mathbb{A}$ and $\mathbb{B}$ are elementarily equivalent (see Section 2.3), that is, they satisfy the same first-order sentences. Indeed, this follows by Theorem 5.8, since every first-order sentence has some finite quantifier rank.

On the other hand, condition (2) rather corresponds to isomorphism, at least for countable structures:

Lemma 6.1. *Let $\mathbb{A}$ and $\mathbb{B}$ be countable relational structures. Then Duplicator wins the ω-round Ehrenfeucht-Fraïssé game on $\mathbb{A}$ and $\mathbb{B}$ if and only if $\mathbb{A} \cong \mathbb{B}$.*

Proof. We show the left-to-right implication, the other one being obvious. Therefore, we construct an isomorphism between $\mathbb{A}$ and $\mathbb{B}$. This construction is called the *back-and-forth* method.

Fix an arbitrary enumeration of the domain of $\mathbb{A}$ and of $\mathbb{B}$. Construct an increasing chain

$$f_0 \subseteq f_0' \subseteq f_1 \subseteq f_1' \subseteq f_2 \subseteq f_2' \ldots \tag{6.1}$$

of finite partial isomorphisms between $\mathbb{A}$ and $\mathbb{B}$ as follows. Let f_0 be the isomorphism with empty domain. Let $i \geqslant 0$ and assume that f_i is already constructed.

Let $a \in \mathbb{A}$ be the first element in the enumeration of $\mathbb{A}$ which is not in the domain of f_i. By Proposition 7.9 (3), the partial isomorphism $f_i \colon \mathbb{A} \rightharpoonup \mathbb{B}$ extends to a partial isomorphism $f_i' \colon \mathbb{A} \rightharpoonup \mathbb{B}$, which is defined on a. Let $b \in \mathbb{B}$ be the first element in the enumeration of $\mathbb{B}$ which is not in the image of f_i'. By a symmetric argument, f_i' extends to a partial isomorphism $f_{i+1} \colon \mathbb{A} \rightharpoonup \mathbb{B}$, such that b is in the image of f_{i+1}. Repeat.

This ends the construction of the chain (6.1). By construction, every element in $a \in \mathbb{A}$ is in the domain of some f_i, and every element $b \in \mathbb{B}$ is in the image of some f_i. For each $a \in \mathbb{A}$, define $g(a)$ as $f_i(a)$, for any i such that $f_i(a)$ is defined. Then $g \colon \mathbb{A} \to \mathbb{B}$ is an isomorphism. ■

To show that the conditions (1) and (2) considered earlier are not equivalent, it is therefore sufficient to show two countable structures $\mathbb{A}$ and $\mathbb{B}$ such that $\mathbb{A}$ and $\mathbb{B}$ are elementarily equivalent, but non-isomorphic. Such examples were given in Exercises 2.16, 2.17, 5.18, and 5.19.

6.2 *Bounded variable fragment*

Fix a number k and k variables $x_1, \ldots, x_k$. The *k-variable fragment* of first-order logic, denoted FO^k is the set of formulas which only use variables $\{x_1, \ldots, x_k\}$.

The key feature of FO^k is that it allows reusing the same variable multiple times.

Example 6.2. For every $n \in \mathbb{N}$ there is a sentence with 3 variables expressing that a given structure is a total order of size at least n.

Note that every formula with l free variables and of quantifier rank at most k is equivalent to a formula with $k + l$ variables. However, formulas with at most k variables can be of arbitrarily large quantifier rank, as the example above demonstrates.

Let $\mathbb{A}, \mathbb{B}$ be structures, $\bar{a} \in \mathbb{A}^l$ and $\bar{b} \in \mathbb{B}^l$ be two tuples of elements of the same length $l \leqslant k$. Write $\bar{a} \equiv_{\mathrm{FO}^k} \bar{b}$ to denote

$$\mathbb{A} \models \varphi(\bar{a}) \quad \Leftrightarrow \quad \mathbb{B} \models \varphi(\bar{b}) \qquad \text{for every } \varphi(x_1, \ldots, x_l) \in \mathrm{FO}^k.$$

Note that contrary to the equivalence $\equiv^k$, the equivalence $\equiv_{\mathrm{FO}^k}$ has infinitely many equivalence classes on l tuples, for a fixed $l \leqslant k$. Indeed, in a total order, the ith smallest element and the jth smallest element are not equivalent with respect to $\equiv_{\mathrm{FO}^2}$, unless $i = j$.

Pebble games. The equivalence $\equiv_{\mathrm{FO}^k}$ can be characterized in terms of games analogous to Ehrenfeucht-Fraïssé games. Note that the number of rounds in the Ehrenfeucht-Fraïssé game corresponds to the quantifier rank of the formula. Hence, as FO^k may have unbounded quantifier rank, intuitively, our games should have an unbounded number of rounds. On the other hand, we should still bound the number of pebbled positions, as those correspond to the variables of a subformula. This leads to the *k-pebble game*, which is played in the same way as the Ehrenfeucht-Fraïssé game, but with the difference that each of the players has only k pebbles (numbered $1, \ldots, k$) at disposal. However, those pebbles can be moved during the game.

A more precise definition is as follows. Let $\mathbb{A}$ and $\mathbb{B}$ be two structures. A position of the game consists of two tuples $\bar{a} \in \mathbb{A}^{\bar{y}}$ and $\bar{b} \in \mathbb{B}^{\bar{y}}$, where $\bar{y} \subseteq \{x_1, \ldots, x_l\}$.

Fix a position $(\bar{a}, \bar{b})$. Consider the relation which relates $\bar{a}(y) \in \mathbb{A}$ to $\bar{b}(y) \in \mathbb{B}$, for $y \in \bar{y}$. If this relation is not the graph of a partial isomorphism from $\mathbb{A}$ to $\mathbb{B}$ then duplicator loses the game. Otherwise, spoiler picks a variable $x \in \bar{x}$ and one of the structures $\mathbb{A}$ and $\mathbb{B}$. Suppose spoiler picks $\mathbb{A}$, the other case is analogous. Spoiler picks an element $c \in \mathbb{A}$ and duplicator picks an element $d \in \mathbb{B}$. The game continues from the position $(\bar{a}[x/c], \bar{b}[x/d])$, where $\bar{a}[x/c] \in \mathbb{A}^{\bar{y}x}$ is the tuple obtained from $\bar{a}$ by setting the coordinate x to c, and $\bar{b}[x/d]$ is analogous.

The game continues until duplicator loses, or continues indefinitely; in this case, duplicator wins.

Example 6.3. Let $\mathbb{A} = \mathbb{B}$ be the directed path of length 1000 and let a, b be two distinct elements. Then spoiler wins the 2-pebble game starting from position (a, b), by using the following strategy: in round i, place pebble number $i \mod 2$ at the successor of the element on which pebble number $(i+1) \mod 2$ is placed.

Example 6.4. Let $\mathbb{A}$ and $\mathbb{B}$ be sets of size 10 and 11, respectively, and let $a \in \mathbb{A}$

and $b \in \mathbb{B}$. Then duplicator wins the 2-pebble game starting from position (a, b).

Theorem 6.5. *Fix a finite set of variables $\bar{x} = \{x_1, \ldots, x_k\}$. Let $\mathbb{A}, \mathbb{B}$ be finite structures, $\bar{a} \in \mathbb{A}^{\bar{y}}, \bar{b} \in \mathbb{B}^{\bar{y}}$ be tuples with $\bar{y} \subseteq \bar{x}$. The following conditions are equivalent:*

1. *$\bar{a} \equiv_{\mathrm{FO}^k} \bar{b}$,*
2. *duplicator has a winning strategy in the k-pebble game starting from position $(\bar{a}, \bar{b})$.*

The implication (2)→(1) holds for arbitrary (not necessarily finite) $\mathbb{A}$ and $\mathbb{B}$.

We will prove the following, stronger statement. Say that a formula $\varphi(\bar{x})$ *distiniguishes* $\bar{a} \in \mathbb{A}^{\bar{y}}$ from $\bar{b} \in \mathbb{B}^{\bar{y}}$ (where $\bar{y} \subseteq \bar{x}$) if $\mathbb{A} \models \varphi(\bar{a})$ and $\mathbb{B} \models \neg\varphi(\bar{b})$.

Lemma 6.6. *Let $\mathbb{A}, \mathbb{B}, \bar{x}, \bar{a}, \bar{b}$ be as in the theorem and $q \in \mathbb{N}$. The following conditions are equivalent:*

1. *there is a formula with variables $\bar{x}$ and of quantifier rank q that distinguishes $\bar{a}$ from $\bar{b}$,*
2. *spoiler has a strategy in the k-pebble game starting from position $(\bar{a}, \bar{b})$ that wins in at most q rounds.*

The implication (1)→(2) holds for arbitrary (not necessarily finite) $\mathbb{A}$ and $\mathbb{B}$.

Lemma 6.6 immediately yields Theorem 6.5, by contrapositive. So we now prove the lemma.

Proof. (1)→(2) We proceed by induction on q. Suppose there is a formula $\varphi(\bar{y}) \in \mathrm{FO}^k$ of quantifier rank q which distinguishes $\bar{a}$ from $\bar{b}$.

The formula $\varphi(\bar{y})$ is equivalent to a Boolean combination of formulas $\varphi'(\bar{x})$ of the form $\exists x.\psi$ where $\psi \in \mathrm{FO}^k$ has quantifier rank smaller than q, or of atomic formulas. As φ distinguishes $\bar{a}$ from $\bar{b}$, there must be a formula $\varphi'(\bar{x})$ of one of those forms such that $\varphi(\bar{x})$ or $\neg\varphi(\bar{x})$ distinguishes $\bar{a}$ from $\bar{b}$. If an atomic formula distinguishes $\bar{a}$ from $\bar{b}$ then spoiler wins immediately, that is, in 0 rounds. So assume $\exists x.\psi$ distinguishes $\bar{a}$ from $\bar{b}$ for some formula $\psi \in \mathrm{FO}^k$ (otherwise we replace the roles of $\bar{a}$ and $\bar{b}$). Then $\bar{a}$ satisfies $\exists x.\psi$ so there is a $c \in \mathbb{A}$ such that $\bar{a}[x/c]$ satisfies ψ and for every $d \in \mathbb{B}$, $\bar{b}[x/d]$ does not satisfy ψ.

Now spoiler picks the structure $\mathbb{A}$ and the variable x, and places the pebble x on the element c of $\mathbb{A}$. From the above, whatever response $d \in \mathbb{B}$ duplicator picks, the formula $\psi \in \mathrm{FO}^k$ distinguishes the resulting tuples, namely $\bar{a}[x/c]$ and $\bar{b}[x/d]$. Since ψ has quantifier rank smaller than q, by inductive assumption, spoiler has a winning strategy that wins in less than q rounds from the new configuration.

(2)$\rightarrow$(1) We proceed by induction on q. The base case $q = 0$ is immediate.

In the inductive step, suppose spoiler wins in q rounds by placing pebble $x \in \bar{x}$ on the vertex $c \in \mathbb{A}$ in the structure $\mathbb{A}$. In particular, for every $d \in \mathbb{B}$, spoiler wins from the position $(\bar{a}[x/c], \bar{b}[x/d])$ in $q-1$ rounds. By inductive assumption, for each $d \in \mathbb{B}$ there is some formula $\psi_d(\bar{y}x)$ in FO^k of quantifier rank at most $q-1$, which is satisfied by $\bar{a}[x/c]$ and not by $\bar{b}[x/d]$. Then $\bar{a}$ satisfies the formula

$$\exists x. \bigwedge_{d \in \mathbb{B}} \psi_d(\bar{y}x),$$

which is a finite formula as $\mathbb{B}$ is finite, and belongs to FO^k, and has quantifier rank at most $q-1$. On the other hand, $\bar{b}$ does not satisfy this formula in $\mathbb{B}$. ■

Exercise 6.7. Show that the equivalence in Theorem 6.5 fails if $\mathbb{A}$ and $\mathbb{B}$ are potentially infinite. For the equivalence to work, one can extend FO^k to the infinitary logic allowing infinite conjunctions and disjunctions, but still restricted to k variables, and formulas of finite quantifier rank.

Exercise 6.8. Fix the signature with one binary relation symbol $\leqslant$. Prove that there is no sentence φ with 2 variables such that for every structure $\mathbb{A}$, the sentence φ holds in $\mathbb{A}$ if and only if $\leqslant$ defines a partial order in $\mathbb{A}$.

Exercise 6.9. Does the statement of the previous exercise still hold if we restrict attention only to finite structures $\mathbb{A}$?

6.3 Counting extension*

We consider an extension of first-order logic by counting quantifiers $\exists^{\geqslant i} x.\varphi$, for all $i \in \mathbb{N}$, expressing that there are at least i elements satisfying a given formula. More precisely, a tuple $\bar{a} \in \mathbb{A}^{\bar{y}}$ satisfies the formula $\exists^{\geqslant i} x.\varphi(\bar{y}x)$, where $\varphi(\bar{y}x)$ is a formula and $i \in \mathbb{N}$ is fixed, if and only if there are at least i distinct elements $c \in \mathbb{A}$ such that $\bar{a}[x/c]$ satisfies the formula $\varphi(\bar{y}x)$.

It is not difficult to see that first-order logic extended with counting quantifiers is no more expressive than first-order logic, since $\exists^{\geqslant i} x.\varphi$ is equivalent to a first-order formula. However, this conversion introduces i quantifiers and variables, so increases both the quantifier rank of the formula, as well as the total number of used variables.

Fix $k \in \mathbb{N}$ and a set of k variables $\bar{x} = \{x_1, \ldots, x_k\}$. Let C^k be the set of formulas with counting quantifiers which only use variables in $\bar{x}$. For two tuples $\bar{a} \in \mathbb{A}^{\bar{y}}$ and $\bar{b} \in \mathbb{B}^{\bar{y}}$ where $\bar{y} \subseteq \bar{x}$, write $\bar{a} \equiv_{\mathrm{C}^k} \bar{b}$ if $\bar{a}$ and $\bar{b}$ satisfy exactly the same formulas $\varphi(\bar{y}) \in \mathrm{C}^k$.

We now give a game characterization of the equivalence $\equiv_{\mathrm{C}^k}$. The game is called the *bijective k-pebble game*. It is played as the k-pebble game, where the round at position $(\bar{a}, \bar{b})$ with $\bar{a} \in \mathbb{A}^{\bar{y}}$ and $\bar{b} \in \mathbb{B}^{\bar{y}}$ has the following form:

1. spoiler picks a variable $x \in \bar{x}$,
2. duplicator picks a bijection f between (the domains of) $\mathbb{A}$ and $\mathbb{B}$,
3. spoiler picks a pair $(c, d) \in \mathbb{A} \times \mathbb{B}$ of elements related by f,
4. the game continues from the position $(\bar{a}[x/c], \bar{b}[x/d])$.

Our goal is to prove that $\bar{a} \equiv_{\mathrm{C}^k} \bar{b}$ if and only if duplicator has a winning strategy in the bijective k-pebble game starting from position $(\bar{a}, \bar{b})$. To prove this, it will be convenient to define yet another characterisation of this condition. As a byproduct, we will obtain a simple polynomial-time algorithm deciding $\equiv_{\mathrm{C}^k}$.

*omitted in the lectures

Weisfeiler-Leman algorithm

We now describe an algorithm deciding whether two k-tuples $\bar{a} \in \mathbb{A}^k, \bar{b} \in \mathbb{B}^k$ satisfy $\bar{a} \equiv_{C^k} \bar{b}$.

The *k-dimensional Weisfeiler-Leman algorithm* (k-WL) is run on a single structure $\mathbb{A}$, and proceeds by coloring tuples in $\mathbb{A}^k$. For $i \in \{1, \dots, k\}$ say that two tuples $\bar{a}, \bar{a}'$ are i-neighbors if $\bar{a}(j) = \bar{a}'(j)$ for all $j \in \{1, \dots, k\} - \{i\}$.

Initially, two tuples $\bar{a}, \bar{b} \in \mathbb{A}^k$ get the same color if and only if there is a partial isomorphism from $\mathbb{A}$ to $\mathbb{A}$ mapping $\bar{a}$ to $\bar{b}$ (that is, $\bar{a}$ and $\bar{b}$ have equal atomic types). In each round, the coloring is refined by assigning different colors to tuples $\bar{a}$ and $\bar{b}$ if there is $i \in \{1, \dots, k\}$ and a color c such that $\bar{a}$ and $\bar{b}$ have a different number of i-neighbors of color c. The algorithm stops when the partition of $\mathbb{A}^k$ into color classes is the same partition as in the one from the previous step (indeed, performing further steps would not lead to any new refinements). The resulting coloring is called the *stable* coloring of $\mathbb{A}^k$. Note that the stable coloring is reached after at most $\mathcal{O}(|\mathbb{A}|^k)$ steps.

Given two structures $\mathbb{A}$ and $\mathbb{B}$ and tuples $\bar{a} \in \mathbb{A}$ and $\bar{b} \in \mathbb{B}$ we may also run k-WL simultaneously on both structures, in each step maintaining a correspondence between the color classes in $\mathbb{A}^k$ and the corresponding color classes in $\mathbb{B}^k$. Equivalently, one could consider running k-WL on the disjoint union of $\mathbb{A}$ and $\mathbb{B}$, but disregarding "mixed" tuples which contain vertices both from $\mathbb{A}$ and from $\mathbb{B}$.

Say that k-WL *does not distinguish* $\bar{a} \in \mathbb{A}^k$ and $\bar{b} \in \mathbb{B}^k$ if they have the same colors in the stable coloring. More generally, if $\bar{a} \in \mathbb{A}^l$ and $\bar{b} \in \mathbb{B}^l$ are two tuples of length $l \leqslant k$, then say that k-WL distinguishes $\bar{a}$ and $\bar{b}$ if in the stable coloring of $\mathbb{A}^k$ and $\mathbb{B}^k$ there is some color class C such that the number of extensions of $\bar{a}$ to a tuple in C is different than the number of extensions of $\bar{b}$ to a tuple in C. In particular, taking $l = 0$, k-WL *distinguishes* the two structures $\mathbb{A}, \mathbb{B}$ if in the stable coloring, for some color class c the structures $\mathbb{A}$ and $\mathbb{B}$ have a different number of tuples of color c. Say that k-WL *identifies* a structure $\mathbb{A}$ if it distinguishes $\mathbb{A}$ from every structure not isomorphic to $\mathbb{A}$.

Example 6.10. 2-WL is known as *the color refinement algorithm*. It does not distinguish two d-regular graphs with the same number of vertices. It identifies all trees.

We are now ready to formulate the equivalence between bijective k-pebble games, counting logic equivalence and k-WL equivalence.

Theorem 6.11. *Fix $k \in \mathbb{N}$, let $\mathbb{A}, \mathbb{B}$ be finite structures, and $\bar{a} \in \mathbb{A}^k, \bar{b} \in \mathbb{B}^k$ be tuples. The following conditions are equivalent:*

1. *$\bar{a} \equiv_{\mathsf{C}^k} \bar{b}$,*
2. *k-WL does not distinguish $\bar{a}$ and $\bar{b}$,*
3. *duplicator has a winning strategy in the bijective k-pebble game starting from the position $(\bar{a}, \bar{b})$.*

Corollary 6.12. *k-WL does not distinguish $\mathbb{A}$ and $\mathbb{B}$ if and only if $\mathbb{A} \equiv_{\mathsf{C}^k} \mathbb{B}$.*

Proof of Theorem 6.11. We prove $(1) \to (2) \to (3) \to (1)$.

$(1) \to (2)$. By induction on the number r of rounds we show that in the coloring reached in the rth round, for each color class C there is a formula $\varphi_C(\bar{x})$ of C^k which holds exactly at those tuples which have color C. The induction base is trivial, as the colors correspond to atomic types with variables $\bar{x}$. In the inductive step, each color class C is defined as a color class C' from the previous step intersected with conditions of the form "the number of i-neighbors in color class D is equal to p" for some $i \in \{1, \ldots, k\}, p \in \mathbb{N}$ and color classes C' and D which we can define using formulas of C^k by inductive assumption. This yields a formula φ_C defining the color class C, finishing the inductive step.

In particular, for every color class C in the stable coloring we have a formula φ_C. Hence, if $\bar{a}$ has color C then $\mathbb{A} \models \varphi_C(\bar{a})$ and by (1), also $\mathbb{B} \models \varphi_C(\bar{b})$, so $\bar{b}$ also has color C, proving (2).

(2)$\to$(3). Duplicator maintains the strategy that at any position $(\bar{a}, \bar{b})$ in the game, $\bar{a}$ and $\bar{b}$ have the same colors in the stable coloring. At the initial position $(\bar{a}, \bar{b})$ this holds by the assumption (2). In the inductive step, suppose spoiler declares he will move pebble $x \in \bar{x}$.

As $\bar{a}$ and $\bar{b}$ have the same colors in the stable coloring, in particular, they have equal numbers of x-neighbors in each color class. It follows that there is a bijection $f\colon \mathbb{A} \to \mathbb{B}$ such that for every $c \in \mathbb{A}$, $\bar{a}[x/c]$ and $\bar{b}[x/f(c)]$ have the same colors in the stable coloring. In particular, $\bar{a}[x/c]$ and $\bar{b}[x/f(c)]$ have the same atomic types, as the coloring refines the partition into atomic types. Hence, such a bijection will not be losing for duplicator, and will maintain the invariant.

(3)→(1). This implication is proved analogously as the implication (2)→(1) in Theorem 6.5, so we omit the proof. ∎

7
Quantifier Elimination

In order to analyze the expressive power of a logic in a given structure $\mathbb{A}$, a basic method used both in model theory and in finite model theory is to prove that it has *quantifier elimination*: every formula $\varphi(\bar{x})$ is equivalent in $\mathbb{A}$ to some quantifier-free formula $\psi(\bar{x})$.

Example 7.1.

- $(\mathbb{N}, =)$ has quantifier elimination. Here's a sketch of an argument. Pick for example $(\mathbb{N}, =)$ and a formula $\varphi(x, y, z)$ with three free variables; we want to show that φ is equivalent to some quantifier-free formula. Let's look at all the triples of elements of $(\mathbb{N}, =)$. There are 4 atomic types of triples, namely:

  $$(x \neq y \neq z \neq x), (x = y \neq z), (x = z \neq y), (y = z \neq x).$$

  Now, observe that for every two triples $\bar{a}$, $\bar{b}$ with the same atomic type, there is some bijection $f\colon \mathbb{N} \to \mathbb{N}$ mapping $\bar{a}$ to $\bar{b}$ componentwise. For example, for the triples $\bar{a} = (1, 2, 5)$ and $\bar{b} = (2, 12, 1500)$, there is even such a bijection f which is the identity on $\mathbb{N} - \{1, 2, 5, 12, 1500\}$. As every bijection from $\mathbb{N}$ to $\mathbb{N}$ is an automorphism of $(\mathbb{N}, =)$, f is an automorphism mapping $\bar{a}$ to $\bar{b}$. It follows that $\mathbb{A} \models \varphi(\bar{a})$ if and only if $\mathbb{A} \models \varphi(\bar{b})$ (see Example 2.5). Hence, the truth value of φ is constant on each of the 4 atomic types: for each atomic type $\tau(x, y, z)$, either φ holds for all triples satisfying τ, or φ holds for no triples satisfying τ, that is,

$\mathbb{A} \models \varphi \to \tau$ or $\mathbb{A} \models \varphi \to \neg\tau$. Hence, $\varphi(x, y, z)$ is equivalent (in $\mathbb{A}$) to the disjuction $\bigvee_\tau \tau(x, y, z)$ ranging over those atomic types $\tau(x, y, z)$ such that $\mathbb{A} \models \varphi \to \tau$. As there are only four atomic types of triples, this is a finite disjunction, and hence a quantifier-free formula.

The same argument works for a formula with k free variables, as we have that for any two k tuples with the same atomic there is an automorphism of $(\mathbb{N}, =)$ mapping one tuple to the other.

- The same argument as above works for the structure $(\mathbb{Q}, \leqslant)$. Here, the automorphisms are the increasing bijections from $\mathbb{Q}$ to $\mathbb{Q}$. It remains to verify that for any two k tuples with the same atomic type there is an automorphism of $(\mathbb{Q}, \leqslant)$ mapping one tuple to the other.
- More generally, suppose that $\mathbb{A}$ is a structure over a finite relational signature. If for every tuples $\bar{a}$ and $\bar{b}$ in $\mathbb{A}$ with the same length and atomic type there is an automorphism of $\mathbb{A}$ mapping $\bar{a}$ to $\bar{b}$ componentwise, then $\mathbb{A}$ has quantifier elimination. The converse implication also holds, if $\mathbb{A}$ is countable (see Lemma 7.19).
- $(\mathbb{N}, \leqslant)$ does not have quantifier elimination, since the formula $\varphi(x) = \exists y. y < x$ expressing that x is larger than 0 is not expressible by a quantifier-free formula. However, the structure $(\mathbb{N}, \leqslant, 0, s)$, where $s \colon \mathbb{N} \to \mathbb{N}$ is the successor function, has quantifier elimination.
- $(\mathbb{R}, +, \cdot, \leqslant)$ has quantifier elimination – this is a result of Alfred Tarski, and is proved in Section 7.1 below,
- $(\mathbb{C}, +, \cdot)$ has quantifier elimination – again a result of Tarski,
- $(\mathbb{Z}, +, \leqslant)$ does not have quantifier elimination, since the property div_n of being divisible by a fixed integer n can be expressed by a first-order formula (e.g. div_2 is definable by $\exists y. y + y = x$), but not by a quantifier-free formula. However, $(\mathbb{Z}, +, \leqslant, (div_n)_{n \geqslant 2})$ has quantifier elimination. This is a result of Mojżesz Presburger, Tarski's student.

As we see above, it is often the case that the structure needs first to be expanded with some definable relations or functions in order to get quantifier elimination. This is for example the case in the structure $(\mathbb{R}, +, \cdot)$. The relation $\leqslant$ is definable by

$$x \leqslant y \equiv \exists z.\, y = x + z \cdot z,$$

but there is no quantifier-free formula using only $+$ and $\cdot$ defining $\leqslant$. However, once the relation $\leqslant$ is added to the structure, we get quantifier elimination.

More generally, we call any result of the form: over a class of structures $\mathcal{C}$, every first-order formula is equivalent to a formula of a specific shape a *quantifier elimination result*.

Exercise 7.2. Show that every quantifier-free formula $\varphi(x)$ using equality $=, +, \cdot$ and arbitrary constants from $\mathbb{R}$ defines a subset of $\mathbb{R}$ which is finite, or whose complement is finite. Conclude that $\leqslant$ cannot be defined by a quantifier-free formula using $+, \cdot, =$ and constants only.

The following lemma gives a simple criterion for quantifier elimination. It says that it is enough to eliminate quantifiers from formulas with a single existential quantifier.

Lemma 7.3. *Let $\mathbb{A}$ be a structure over a signature Σ. Then $\mathbb{A}$ has quantifier elimination if and only if every formula of the form $\exists y.\alpha(\bar{x}y)$, where α is a conjunction of literals, is equivalent to a quantifier-free formula in $\mathbb{A}$.*

Proof. Given a formula $\varphi(\bar{x})$ we eliminate the quantifiers from inside out. More formally, we proceed by induction on the structure of the formula φ. In the base case, φ is an atomic formula and there is nothing to prove. In the inductive step, φ is of the form $\neg\alpha$, or of the form $\alpha \vee \beta$, or is of the form $\exists y.\psi(\bar{x}y)$, for some formula ψ. In the first two cases, α and β may be assumed to be quantifier-free by inductive assumption, and we are done. So consider the third case, and by inductive assumption, we may assume that $\psi(\bar{x}y)$ is a quantifier-free formula. Rewrite $\psi(\bar{x}y)$ into disjunctive normal form, as a disjunction of formulas $\psi_i(\bar{x}y)$, each of which is a conjunction of literals. Now, $\exists y.\psi(\bar{x}y)$ is equivalent to the disjunction of $\exists y.\psi_i(\bar{x}y)$ which, by assumption, is equivalent to a quantifier-free formula. ∎

7.1 Tarski Arithmetic[*]

In this section we prove the result of Tarski.

Theorem 7.4 (Tarski, 1931). *The ordered field of reals $(\mathbb{R}, +, \cdot, \leqslant)$ has quantifier-elimination. Moreover, there is an algorithm that, given a formula $\varphi(x_1, \dots, x_k)$ computes a quantifier-free formula $\varphi'(x_1, \dots, x_k)$ equivalent to $\varphi(x_1, \dots, x_k)$ in $(\mathbb{R}, +, \cdot, \leqslant)$.*

Let us briefly discuss some consequences of the theorem. Consider the formula $\varphi_2(a, b, c) \equiv \exists x. ax^2 + bx + c = 0$ with free variables a, b, c. The theorem implies that this formula is equivalent to a quantifier-free formula, and indeed, we know well that $\varphi_2(a, b, c)$ is equivalent to the formula $b^2 - 4ac \geqslant 0$. We may consider a similar formula $\varphi_k(a_0, \dots, a_k) \equiv \exists x. \sum_{i=0}^{k} a_i x^{k-i}$, and Tarski's theorem in particular tells us that there is some quantifier-free formula $\alpha_k(a_0, \dots, a_k)$ that is equivalent to φ_k. So we may think of the formula α_k a generalized form of the discriminant, for an arbitrary degree k. This formula allows us, in particular, to determine whether or not a given polynomial of degree k has a real root, using a constant number of arithmetic operations and comparisons using $\geqslant$. Tarski's theorem also implies that we can determine the number of real roots of a given polynomial of degree k. Indeed, to determine whether a polynomial $p(x)$ has at least k roots, determine whether or not the sentence $\exists x_1, \dots, x_k. (x_1 < x_2 < \dots < x_k) \wedge (p(x_1) = \dots = p(x_k) = 0)$ holds in $(\mathbb{R}, +, \cdot, \leqslant)$. It follows from Tarski's result that this can be done effectively. Furthermore, for any $j, k \in \mathbb{N}$ we may compute a quantifier-free formula $\alpha_k^j(a_0, \dots, a_k)$ that holds of a tuple $(a_0, \dots, a_k)$ if and only if the polynomial $\exists x. \sum_{i=0}^{k} a_i x^{k-i}$ has at least (or exactly) j distinct roots.

Before proving the theorem, we formulate and prove a key lemma underlying it. On the face of it, it does not concern logic, but solutions to systems of polynomial equations and inequalities with one variable. The lemma states roughly that if we are given a finite set of polynomials P with one free variable then we can find, in a finite number of steps, a finite partition $\mathcal{P}$ of $\mathbb{R}$ into intervals and singletons such that in each part of the partition, each polynomial in P has constant sign. This is formalized as follows.

[*]omitted in the lectures

Let P be a finite family P of polynomials $p = p(x)$ with one free variable, denoted x, and with real coefficients. An *analysis* of P is a partition $\mathcal{P}$ of $\mathbb{R}$ into finitely many open intervals (possibly unbounded) and singletons, together with a function $A\colon \mathcal{P} \times P \to \{-, 0, +\}$ such that for every $U \in \mathcal{P}$ and polynomial $p(x) \in P$,

- if $A(U, p) = +$ then $p(x) > 0$ for all $x \in U$,
- if $A(U, p) = -$ then $p(x) < 0$ for all $x \in U$,
- if $A(U, p) = 0$ then $p(x) = 0$ for all $x \in U$.

An analysis is simply denoted A, and its *size* is the size of $|\mathcal{P}|$. Note that if P is a family of k polynomials of degree at most d each, then P has an analysis A of size at most $2k \cdot d + 1$. Indeed, the set Z that is the union of the set of real roots of all polynomials $p(x)$ in P has size at most $k \cdot d$, and the partition $\mathcal{P}$ partitions $\mathbb{R}$ by cutting at each element of Z, that is, $\mathcal{P}$ consists of all singletons $\{z\}$ for $z \in Z$ and all intervals (u, v) where u, v are two consecutive elements of $\{-\infty, +\infty\} \cup Z$, ordered naturally.

Note that if $P \subseteq Q$ are two sets of polynomials then any analysis of Q induces an analysis of P, of the same size. An analysis is *irredundant* if for every singleton $\{c\}$ occuring in $\mathcal{P}$, there is some $p \in P$ such that $p(c) = 0$.

We now describe an *algorithm* that, for any given family P of polynomials $p(x)$ with real coefficients computes an (irredundant) analysis A of P. The algorithm performs a number of steps bounded in terms of the number of polynomials in P and their maximum degree. We will not go into the details of the computation model in which the algorithm is performed. We assume that it is a program that can access the coefficients of the polynomials in P, can manipulate real numbers with infinite precision and perform the operations $+$ and $\cdot$ on them, as well as test if a given number is > 0, < 0 or $= 0$. The point is that the run of the algoritm on any given family P of polynomials (given as a $|P| \times d$-matrix of coefficients whose (i, j) entry is the coefficient at x^j of the ith polynomial in P) can be described as a branching decision diagram of bounded (in terms of $|P|$ and d) size, whose inner nodes are labeled by comparisons of the form $E > 0$, where E is an expression using $+, -, \cdot$ and the coefficients in P, and leaf nodes are labeled by the output analysis A.

An output analysis A is represented *symbolically* (without explicitly representing the endpoints of the intervals in $\mathcal{P}$), by specifying:

- the size m of the partition $\mathcal{P}$ underlying A,
- a function $\hat{A}\colon \{1, \ldots, m\} \times P \to \{-, 0, +\}$,

such that $\hat{A}(i, p) = A(U_i, p)$ where U_i is the ith smallest interval/singleton in the partition $\mathcal{P} = \{U_1, \ldots, U_m\}$ (ordered naturally), for $i = 1, \ldots, m$. Note that U_i is an interval if i is odd and is a singleton if i is even, and that m is odd.

Lemma 7.5. *For every fixed k (number of polynomials) and d (maximum degree) there is an algorithm as above that inputs a family P of k polynomials of degree at most d, and outputs an analysis of P. The number of steps performed by the algorithm is bounded in terms of k, $|P|$ and d.*

Proof. We give a rought sketch of the algorithm and leave to the reader the verification that this algorithm can indeed be carried out using a algorithm manipulating real numbers, as described above.

Let P be given on input. Close the set P under taking derivatives: if $p \in P$ then $p' \in P$. Next, close the set P under taking remainders under polynomial division: if p, q are two polynomials in P then there is a polynomial $r \in P$ such that $p = q \cdot s + r$ for some polynomial s. To close P under taking remainders we perform the Euclidean algorithm for division of polynomials; this can be computed in our computation model, given $\bar{a}$. Repeat those two steps until obtaining a set of polynomials P that is closed both under taking derivatives and remainders. Note that the number of steps can be bounded in terms of k and d, since closing under remainders and derivatives only adds polynomials of lower degree.

Hence, after performing a bounded number of steps, we may assume that P is closed under taking derivatives and remainders. We are now ready to compute an analysis of P.

Let $p_1, p_2, \ldots, p_m$ be the polynomials in P, sorted according to their degree, so that p_1 has the lowest degree and p_m has the highest degree.

For $i = 1, \ldots, m$ we compute an (irredundant) analysis of $\{p_1, \ldots, p_{i+1}\}$, as follows. Suppose we have already computed an analysis A of $\{p_1, \ldots, p_i\}$ for some $i \geqslant 0$, which will be now used to compute an analysis A' of $\{p_1, \ldots, p_{i+1}\}$. Let $\mathcal{P}$ be the underlying partition of $\mathbb{R}$ (represented symbolically, by its length). Denote $p = p_{i+1}$. Suppose that p is non-constant, since otherwise the analysis A' is trivial to obtain, by checking the sign of the constant appearing in p. Note that $p'(x)$ has lower degree than $p(x)$, so p' is among $p_1, \ldots, p_i$.

We create a new partition $\mathcal{P}'$ and analysis A' basing on $\mathcal{P}$ and A. Initially, $\mathcal{P}' = \mathcal{P}$. We will then refine the partition $\mathcal{P}'$. Note that if $\mathcal{P}'$ is a refinement of $\mathcal{P}$ then the value $A'(U, q)$, for $q \in \{p_1, \ldots, p_i\}$ and $U \in \mathcal{P}'$, is determined easily from the value $A(V, q)$, where $V \in \mathcal{P}$ is unique such that $U \subseteq V$. Hence, it will only be necessary to determine the values $A'(U, p)$ for $p = p_{i+1}$ and all $U \in \mathcal{P}'$.

For each interval $U \in \mathcal{P}$ perform the following. Let u_1, u_2 be the endpoints of U (possibly infinite). Since A is irredundant, there is a polynomial $q_1 \in \{p_1, \ldots, p_i\}$ which has a root at u_1, and similarly, there is a polynomial $q_2 \in \{p_1, \ldots, p_i\}$ that has a root at u_2. Let r_1 be the remainder of the division of p by q_1. Then r_1 has lower degree than p, so is among $p_1, \ldots, p_i$. Also, $p(x) = s(x) \cdot q_1(x) + r_1(x)$ for some polynomial $s(x)$. Since $q_1(u_1) = 0$ we have that $p(u_1) = r_1(u_1)$. Hence, the sign of $p(u_1)$ is the same as the sign of $r_1(u_1)$, and therefore is determined uniquely by the analysis A. Similarly, we can determine the sign of $p(u_2)$. This way, we determine the values $A'(\{u_1\}, p)$ and $A'(\{u_2\}, p)$.

We now need to figure out the behaviour of p inside the interval $U = (u_1, u_2)$. First observe that there is no root of $p'(x)$ inside the interval U, since p' is among $p_1, \ldots, p_i$, so such a root c would imply that $\{c\}$ belongs to $\mathcal{P}$, which cannot occur since $\mathcal{P}$ is a partition and $c \in U \in \mathcal{P}$. As p' has no root in U, this implies that $p(x)$ has no local extremum in U.

We now consider two cases.

Case 1: $p(u_1)$ and $p(u_2)$ do not have opposite signs. Suppose this sign is nonnegative both for $p(u_1)$ and for $p(u_2)$. Then $p(x) > 0$ for all $x \in U$, since otherwise, by the extreme value theorem, p would have a local minimum in the interval U.

So we set $A'(U, p)$ to $+$ in this case. Similarly, if $p(u_1)$ and $p(u_2)$ are both nonpositive, we set $A'(U, p)$ to $-$.

Case 2: $p(u_1)$ and $p(u_2)$ have opposite signs. Then $p(x)$ has a root r inside the interval U, by the intermediate value theorem. Moreover, $p(x)$ cannot have two roots r_1, r_2 inside the interval U, since otherwise it would have a local extremum inside U. Split $U = (u_1, u_2)$ into three parts $(u_1, r), \{r\}, (r, u_2)$. The sign of p on (u_1, r) is the same as the sign of $p(u_1)$, the sign of p on (r, u_2) is the same as the sign of $p(u_2)$, and $p(r) = 0$. In the partition $\mathcal{P}'$ split the interval U into three parts $(u_1, r), \{r\}, (r, u_2)$ and for each of those parts V define $A'(V, p)$ as discussed.

After having performed this operation for each interval $U \in \mathcal{P}$, we obtain a refinement $\mathcal{P}'$ of $\mathcal{P}$, and an analysis A' based on this refinement. ■

Proof. We show that the condition in Lemma 7.3 holds. Let $\alpha(\bar{y}x)$ be a conjunction of literals, that is, of atomic formulas, or their negations. We show that $\exists y.\alpha(\bar{y}x)$ is equivalent to some quantifier-free formula $\alpha'(\bar{y})$.

Note that $\alpha(\bar{y}x)$ is equivalent to a conjunction of formulas of the form

$$p(\bar{y}x) > 0, \quad p(\bar{y}x) \geqslant 0, \quad \text{or} \quad p(\bar{y}x) = 0, \tag{7.1}$$

where $p(\bar{y}x)$ is a polynomial with integer coefficients and variables $\bar{y}x$.

For a fixed tuple $\bar{a} \in \mathbb{R}^{\bar{y}}$ let $P_{\bar{a}}$ denote the set of polynomials $p(\bar{a}, x)$ such that one of the conjuncts (7.1) occurs in $\alpha(\bar{y}x)$. Then $P_{\bar{a}}$ is a finite set of polynomials with real coefficients and one variable. Note that if we are given an analysis $A_{\bar{a}}$ of $P_{\bar{a}}$ (represented symbolically), then we can easily decide whether or not $\exists x.\alpha(\bar{a}, x)$ holds. Namely, it is sufficient to verify the existence of such an x in each part U of the partition $\mathcal{P}$ underlying $A_{\bar{a}}$. Note that for each part $U \in \mathcal{P}$, each of the polynomials $p(\bar{a}, x)$ has constant sign, for all $x \in U$, so in fact either $\alpha(\bar{a}, x)$ holds for all $x \in U$, or it holds for no $x \in U$, and which case holds can be easily determined by looking up the values of $A_{\bar{a}}(U, p)$, for all $p(x) \in P_{\bar{a}}$.

Now consider the following algorithm. Given on input the tuple $\bar{a}$, first compute the family $P_{\bar{a}}$. Next, run the algorithm from Lemma 7.5, and finally, given its output analysis $A_{\bar{a}}$, output $\bot$ or $\top$, depending on whether the analysis implies $\exists x.\alpha(\bar{a}, x)$ or its negation, as described above.

This algorithm can be represented by a branching program of size bounded in terms of the maximum degree of a polynomial in P and $|P|$, where each inner node is labeled by a quantifier-free formula with parameters from $\bar{a}$, whereas each leaf is labeled $\bot$ or $\top$.

As this branching program has bounded size, it can be also represented as a quantifier-free formula $\alpha'(\bar{y})$. By construction, $\alpha'(\bar{y})$ is equivalent to $\exists x.\alpha(\bar{y}x)$. ∎

Exercise 7.6. Prove that every subset of $\mathbb{R}$ that is definable in $(\mathbb{R}, +, \cdot, \leqslant)$ is a finite union of intervals (possibly unbounded) and single elements.

Exercise 7.7. A real number $r \in \mathbb{R}$ is algebraic if there is a non-zero polynomial $p(x)$ with integer coefficients such that $p(r) = 0$. A real number $r \in \mathbb{R}$ is definable if there is a formula $\varphi(x)$ in the language of $(\mathbb{R}, +, \cdot, \leqslant)$ such that $\mathbb{R} \models \varphi(a)$ if and only if $a = r$, for all $a \in \mathbb{R}$. Prove that a number is algebraic if and only if it is definable.

Exercise 7.8. Show that the set of algebraic numbers forms a field with respect to $+$ and $\cdot$. Moreover, this field, extended with $\leqslant$, has the same theory as $(\mathbb{R}, +, \cdot, \leqslant)$.

7.2 *Relational structures*

In this section, we consider structures over a finite relational signature. In this case, the criterion given by Lemma 7.3 can be further simplified to the case when $\alpha(\bar{x}y)$ is an atomic formula, which in turn can be characterized combinatorially. This yields the following characterization.

Proposition 7.9. *Let $\mathbb{A}$ be a structure over a finite relational signature Σ. The following conditions are equivalent:*

1. *$\mathbb{A}$ has quantifier elimination;*

2. *every formula of the form $\exists y.\alpha(\bar{x}y)$, where α is an atomic type, is equivalent to a quantifier-free formula in $\mathbb{A}$;*

3. *For every finite structure $\widehat{\mathbb{B}}$ which embeds into $\mathbb{A}$ and its induced substructure $\mathbb{B}$ with $|\widehat{\mathbb{B}}| - |\mathbb{B}| = 1$, every embedding $f\colon \mathbb{B} \to \mathbb{A}$ extends to an embedding $\hat{f}\colon \widehat{\mathbb{B}} \to \mathbb{A}$.*

Proof. We show (2)→(1), the other direction being obvious.

By Lemma 7.3, it suffices to eliminate quantifiers from formulas of the form $\exists y.\varphi(\bar{x}y)$ where φ is quantifier-free. Any such formula is equivalent to a finite disjunction of atomic types $\bigvee_i \tau_i$ (cf. Exercise 5.5). Then $\exists y.\varphi(\bar{x}y)$ is equivalent to $\bigvee_i \exists y.\tau_i(\bar{x}y)$, where each τ_i is an atomic type, so it remains to eliminate quantifiers from the formulas $\exists y.\tau(\bar{x}y)$ where τ is an atomic type. Hence, (2) implies quantifier elimination, and is clearly also necessary.

We now show (3)→(2). The converse implication is similar.

Let $\tau(\bar{x}y)$ be an atomic type. If $\tau(\bar{x}y)$ is not satisfiable in $\mathbb{A}$ then $\exists y.\tau(\bar{x}y)$ is equivalent to $\bot$ in $\mathbb{A}$ and we are done. Otherwise, if $y = x$ is a conjunct of $\tau(\bar{x}y)$ for some $x \in \bar{x}$ then $\exists y.\tau(\bar{x}y)$ is equivalent to the formula obtained from $\tau(\bar{x}y)$ by substituting x for y, so we are also done. So assume this is not the case.

We show that $\exists y.\tau(\bar{x}y)$ is equivalent to the atomic type $\sigma(\bar{x})$ obtained by removing from τ all the literals involving y. Clearly, $\exists y.\tau(\bar{x}y) \to \sigma(\bar{x})$ holds in $\mathbb{A}$; we show the other implication. Suppose that $\bar{a} \in \mathbb{A}^{\bar{x}}$ satisfies $\sigma(\bar{x})$. Then condition (3) implies that $\bar{a}$ extends to a tuple $\bar{a}a \in \mathbb{A}^{\bar{x}y}$ which satisfies $\tau(\bar{x}y)$. This is explained in detail below.

Consider the structure $\mathbb{A}_\tau$ corresponding to the atomic type τ (cf. Exercise 5.3). Its domain is a partition of $\bar{x}y$ induced by the equalities appearing in τ. By assumption, y forms a singleton part. Hence, $\mathbb{A}_\sigma$ is the induced substructure of $\mathbb{A}_\tau$ obtained by removing the singleton part of y. Since $\tau(\bar{x}y)$ is satisfiable in $\mathbb{A}$, it follows that $\mathbb{A}_\tau$ embeds into $\mathbb{A}$. The tuple $\bar{a} \in \mathbb{A}^{\bar{x}}$ induces an embedding $f\colon \mathbb{A}_\sigma \to \mathbb{A}$ such that $f([x]) = \bar{a}(x)$ for every $x \in \bar{x}$ and its part $[x] \in \mathbb{A}_\sigma$. By (3), f extends to an embedding $\widehat{f}\colon \mathbb{A}_\tau \to \mathbb{A}$. Let $a = \widehat{f}([y])$. Then $\bar{a}a$ satisfies τ in $\mathbb{A}$, as required. ■

Example 7.10. The following structures have quantifier elimination:

- $(\mathbb{N}, =)$,
- $(\mathbb{Q}, \leqslant)$,
- every structure over a finite signature with unary relations only.

In each case, quantifier elimination can be easily checked by the criterion (3) from Proposition 7.9.

The following lemma says that each countable structure over a finite relational signature which eliminates quantifiers is uniquely determined, up to isomorphism, by the class of its finite induced substructures.

Lemma 7.11. *Let $\mathbb{A}$ and $\mathbb{B}$ be two countable structures over a finite relational signature which both admit quantifier elimination, and which embed the same finite structures. Then $\mathbb{A}$ and $\mathbb{B}$ are isomorphic.*

Proof. We prove that Duplicator wins the ω-round Ehrenfeucht-Fraïssé game on $\mathbb{A}$ and $\mathbb{B}$. The conclusion then follows from Lemma 6.1.

We show that if $a_1, \dots, a_k$ are elements in $\mathbb{A}$ and $b_1, \dots, b_k$ are elements in $\mathbb{B}$ such that

$$\operatorname{atp}_{\mathbb{A}}(a_1, \dots, a_k) = \operatorname{atp}_{\mathbb{B}}(b_1, \dots, b_k). \tag{7.2}$$

then for every $a_{k+1} \in \mathbb{A}$ there is some $b_{k+1} \in \mathbb{B}$ such that

$$\operatorname{atp}_{\mathbb{A}}(a_1, \dots, a_k, a_{k+1}) = \operatorname{atp}_{\mathbb{B}}(b_1, \dots, b_k, b_{k+1}). \tag{7.3}$$

Indeed, let $\widehat{\mathbb{C}}$ be the substructure of $\mathbb{A}$ induced by $a_1, \dots, a_{k+1}$, let $\mathbb{C} \subseteq \widehat{\mathbb{C}}$ be the substructure of $\mathbb{A}$ induced by $a_1, \dots, a_k$, and let $f \colon \mathbb{C} \to \mathbb{B}$ be the function mapping a_i to b_i, for $i = 1, \dots, k$. Then f is an embedding of $\mathbb{C}$ to $\mathbb{B}$, by (7.2). Since $\widehat{\mathbb{C}}$ embeds into $\mathbb{A}$, it is also embeds into $\mathbb{B}$, as $\mathbb{A}$ and $\mathbb{B}$ embed the same finite structures.

By condition (3) in Proposition 7.9, there is an embedding $\widehat{f} \colon \widehat{\mathbb{C}} \to \mathbb{B}$ extending f. Let $b_{k+1} = f(a_{k+1})$. Then (7.3) holds, as $\widehat{f}$ is an embedding. This yields the conclusion.

Using this, we see that Duplicator has a winning strategy in the ω-round Ehrenfeucht-Fraïssé game on $\mathbb{A}$ and $\mathbb{B}$: if in a position $(\mathbb{A}, a_1, \ldots, a_k)$, $(\mathbb{B}, b_1, \ldots, b_k)$ Spoiler places a pebble on a vertex a_{k+1} of $\mathbb{A}$ then Duplicator responds by placing a pebble on the vertex b_{k+1} of $\mathbb{B}$ as obtained above, and does not (immediately) lose by (7.3). If Spoiler places a pebble on a vertex b_{k+1} of $\mathbb{B}$, the argument is symmetric. Hence, Duplicator can always play without losing, so in fact she wins the game with ω rounds. By Lemma 6.1, $\mathbb{A}$ and $\mathbb{B}$ are isomorphic. ■

7.3 *Random graphs*

Fix a set of vertices V. Construct a graph with vertices V randomly, by choosing randomly for each pair of distinct vertices $u, v \in V$ whether to make u and v adjacent. All the choices are made independently and have equal probabilities of the two outcomes (connected/not connected). This process describes a *random graph* with vertices V. It is not a single graph, but a random variable.

For a property of graphs $\mathcal{P}$, let $\Pr[R \in \mathcal{P}]$ denote the probability that R satisfies $\mathcal{P}$. If V is finite, then

$$\Pr[R \in \mathcal{P}] = \frac{|\{G : \textit{graph on } V \textit{ satisfying } \mathcal{P}\}|}{|\{G : \textit{graph on } V\}|}.$$

Denote

$$\Pr[\mathcal{P}] = \lim_{n\to\infty} \Pr[R_n \in \mathcal{P}]$$

where R_n is the random graph on n vertices.

Example 7.12. Let $\mathcal{P}$ be the property of containing a clique on 5 vertices. A random graph on 5 vertices is a clique with probability 2^{-10}. The probability that a random graph on $5n$ vertices does not contain a 5-clique is at most $(1-2^{-10})^n$, which converges to 0 with $n \to \infty$. Hence, $\Pr[\textit{contains a } 5\textit{-clique}] = 1$.

Exercise 7.13. Show:

$$\Pr[\textit{has an isolated vertex}] = 0.$$
$$\Pr[\textit{connected}] = 1.$$
$$\Pr[\textit{has an even number of edges}] = \frac{1}{2}.$$
$\Pr[\textit{has an even number of vertices}]$ does not exist.

For the second item, it may be useful to look at the distance between two fixed vertices.

The main result of this section is the following.

Theorem 7.14. *For any first-order sentence φ, the limit $\Pr[\varphi]$ exists and is either equal to 0 or 1.*

Theorem 7.14 can be used to show inexpressibility results.

Example 7.15. There is no first-order sentence φ such that φ holds in a finite graph G if and only if it has an even number of edges.

Countable random graph. Let R denote the random graph on a fixed countable set V of vertices. Contrary to the case of finite graphs, with probability 1 the resulting graph is always the same (up to isomorphism). This justifies calling R "the" infinite random graph. It is also called the *Rado graph*, after its inventor.

Proposition 7.16. *Let R be the random graph. Then with probability 1, R admits quantifier elimination and embeds every finite graph. In particular, two independent copies of R are isomorphic with probability 1.*

The second part of the statement follows from the first part and from Lemma 7.11.

Before proving Proposition 7.16 we show that R almost surely satisfies the so-called *extension axioms*, as expressed below.

Lemma 7.17. *Let $A, B \subseteq V$ be two disjoint, finite sets of vertices. Then with probability 1 there is a vertex $v \in V$ which is adjacent to all vertices in A and non-adjacent to all vertices in B.*

Proof. The probability that a fixed vertex $v \notin A \cup B$ has the required property is $c := 1/2^{|A|+|B|} > 0$. The probability that some vertex in a set $X \subseteq V - (A \cup B)$ of n vertices has the required property is $1 - (1-c)^n$ and converges to 1 when $n \to \infty$. ■

Proof of Proposition 7.16. It is easy to see that the random graph R contains every finite graph G with probability 1. Indeed, if $|V(G)| = n$ then any fixed n vertices of R form a graph isomorphic to G with probability at least $2^{-\binom{n}{2}}$. Fix $k \geqslant 0$ and let $X_1, \dots, X_k \subseteq \mathbb{N}$ be k pairwise disjoint sets of size n. By independence, the probability that neither of those sets induces a graph isomorphic to G is at most $(1 - 2^{-\binom{n}{2}})^k$. This is an upper bound on the probability p that R does not contain a subgraph isomorphic to G, so

$$p \leqslant (1 - 2^{-\binom{n}{2}})^k, \quad \text{for all } k \geqslant 0.$$

As k is arbitrary and n is fixed, this proves that $p = 0$. Hence, R contains a subgraph isomorphic to G.

We now show that the random graph R almost surely (that is, with probability 1) has quantifier elimination, by proving that condition (3) from Proposition 7.9 holds almost surely in R.

Take any two finite graphs G and $\widehat{G}$ where $\widehat{G}$ extends G by one vertex x and G embeds into R. Partition G into the set of neighbors and non-neighbors of x in $\widehat{G}$. Let $f\colon G \to R$ be an embedding and let A be the image of the set of neighbors of x and B be the image of the set of non-neighbors of x. Let v be as in Lemma 7.17. Then the extension $\widehat{f}$ of f mapping x to v is an embedding of $\widehat{G}$ into R.

One needs to be a bit careful before concluding that condition (3) holds. What we have shown is that for any triple $(G, \widehat{G}, f)$ such that $\widehat{G}$ is a finite graph, G is its subgraph with one vertex removed, and f is an embedding of G into R, the function f can be extended to an embedding of $\widehat{G}$ with probability 1. This is not the same as saying that with probability 1, for all $(G, \widehat{G}, f)$, f can be extended to an embedding of $\widehat{G}$ into R. However, the latter holds as well, since there are only countably many events to consider. This is formalized as follows.

For a triple $(G, \widehat{G}, f)$, where $\widehat{G}$ is a finite graph, G is its subgraph with one vertex removed, and f is a fixed injection from $V(G)$ into $\mathbb{N}$, define the event $\mathcal{E}_{G,\widehat{G},f}$ as the set of all graphs R with vertices $\mathbb{N}$ such that the following implication holds: If f is an embedding from G to R, then there exists an extension $\widehat{f}\colon \widehat{G} \to R$.

By the argument presented above, the event $\mathcal{E}_{G,\widehat{G},f}$ has probability 1. Moreover, there are only countably many events $\mathcal{E}_{G,\widehat{G},f}$ (we may consider only finite graphs $\widehat{G}$ with vertices $1, \dots, n$ for some $n \in \mathbb{N}$). Now, the intersection of a countable family of events of probability 1 is itself an event of probability 1 (by complementation, this is equivalent to saying that a union of countably many sets of measure zero has measure zero). Hence,

$$\Pr[R \in \bigcap_{G,\widehat{G},f} \mathcal{E}_{G,\widehat{G},f}] = 1,$$

where the intersection is over all (countably many) triples $(G, \widehat{G}, f)$ as above. This means that with probability 1, the random graph R satisfies condition (3).

Now, Lemma 7.11 implies that two copies of the random graph are almost surely isomorphic. ∎

Finite random graphs. We will now see that finite random graphs behave much like the infinite random graph, at least when first-order sentences are considered.

For each n, let R_n be a random graph on n vertices, where each edge is drawn independently at random with probability $1/2$.

Theorem 7.18. *For every sentence α, $\lim_{n\to\infty} \Pr[R_n \models \alpha]$ exists and is 1 if the infinite random graph satisfies α, and is 0 otherwise.*

Proof. Let α be a sentence such that $\lim_{n\to\infty} \Pr[R_n \models \alpha]$ does not exist or is smaller than 1. We show that the random graph R satisfies $\neg\alpha$.

Consider the set T of sentences φ such that $\lim_{n\to\infty} \Pr[R_n \models \varphi] = 1$. The same proof as in Lemma 7.17 shows that T includes, for each m, the sentence γ_m "for every two disjoint sets A, B with $|A|, |B| \leqslant m$ there is a vertex v adjacent to A and non-adjacent to B".

By compactness $T \cup \{\neg\alpha\}$ has a model. Indeed, pick any finite set of sentences in T. Clearly their conjunction φ is also in T. By definition of T and the assumption about α, there is a finite graph satisfying $\varphi \wedge \neg\alpha$. By compactness, $T \cup \{\neg\alpha\}$ has a countable model G.

It follows from the proof of Proposition 7.16 that every countable graph satisfying the sentences $\{\gamma_m \mid m \in \mathbb{N}\}$ is isomorphic to the random graph R. In particular, R satisfies $\neg\alpha$.

This proves that if R satisfies α then $\lim_{n\to\infty} \Pr[R_n \models \alpha] = 1$. Conversely, if R satisfies $\neg\alpha$ then $\lim_{n\to\infty} \Pr[R_n \models \neg\alpha] = 1$, and therefore $\lim_{n\to\infty} \Pr[R_n \models \alpha] = 0$. ■

7.4 *Fraïssé limits*[*]

A *partial isomorphism* between $\mathbb{A}$ and $\mathbb{B}$ is an embedding $f\colon \mathbb{A}' \to \mathbb{B}$ of a finite induced substructure $\mathbb{A}' \subseteq \mathbb{A}$ into $\mathbb{B}$. Say that a partial isomorphism is *finite* if its domain is finite. A structure $\mathbb{A}$ such that every finite partial isomorphism between $\mathbb{A}$ and $\mathbb{A}$ extends to an automorphism of $\mathbb{A}$ is called *homogeneous*.

Lemma 7.19. *Let $\mathbb{A}$ be a countable structure over a finite relational signature Σ. Then $\mathbb{A}$ is homogeneous if and only if $\mathbb{A}$ has quantifier elimination.*

Exercise 7.20. Prove Lemma 7.19.

By Lemma 7.11, for any class $\mathcal{C}$ of finite Σ-structures (where Σ is finite relational) such that $\mathcal{C}$ is closed under taking embeddings, if there is a countable homogeneous structure $\mathbb{A}$ such that $\mathcal{C}$ is the class of finite structures which embed into $\mathbb{A}$, then $\mathbb{A}$ is unique, up to isomorphism. The structure $\mathbb{A}$ is then called the *Fraïssé limit* of $\mathcal{C}$.

Example 7.21.
- The Fraïssé limit of the class of finite sets, as structures over the signature with equality only, is $(\mathbb{N}, =)$,
- The Fraïssé limit of the class of finite total orders is $(\mathbb{Q}, \leqslant)$,

[*]omitted in the lectures

- The Fraïssé limit of the class of finite graphs is the (countable) random graph.

Classes $\mathcal{C}$ of finite structures which have a Fraïssé limit can be described purely combinatorially, as follows. Say that $\mathcal{C}$ has the *amalgamation property* if for every $\mathbb{A}, \mathbb{B}_1, \mathbb{B}_2 \in \mathcal{C}$ and embeddings $f_1 \colon \mathbb{A} \to \mathbb{B}_1$ and $f_2 \colon \mathbb{A} \to \mathbb{B}_2$ there exists $\mathbb{C} \in \mathcal{C}$ and embeddings $g_1 \colon \mathbb{B}_1 \to \mathbb{C}$ and $g_2 \colon \mathbb{B}_2 \to \mathbb{C}$ such that $g_1 \circ f_1 = g_2 \circ f_2$, that is, the following diagram commutes:

$$\begin{tikzcd}
 & \mathbb{A} \arrow[dl, "f_1"'] \arrow[dr, "f_2"] & \\
\mathbb{B}_1 \arrow[dr, "g_1"'] & & \mathbb{B}_2 \arrow[dl, "g_2"] \\
 & \mathbb{C} &
\end{tikzcd}$$

Exercise 7.22. Show that the following classes have the amalgamation property:

- the class of all finite total orders,
- the class of all finite graphs,
- the class of all finite posets,
- the class of all finite triangle-free graphs.

Theorem 7.23. *Let Σ be a finite relational signature without relations of arity 0 and let $\mathcal{C}$ be a class of finite structures which is closed under embeddings. Then the following conditions are equivalent:*

- *there exists a countable homogeneous structure $\mathbb{A}$ such that $\mathcal{C}$ is the class of finite structures which embed into $\mathbb{A}$,*
- *$\mathcal{C}$ has the amalgamation property.*

Moreover, the structure $\mathbb{A}$ as above is unique, up to isomorphism.

Exercise 7.24. Prove the top-down implication in Theorem 7.23.

Proof sketch. We sketch the bottom-up implication. The aim is to construct a countable structure $\mathbb{A}$ satisfying the following property $(*)$:

> *for every* $\widehat{\mathbb{B}} \in \mathcal{C}$ *and its induced substructure* $\mathbb{B}$ *with* $|\widehat{\mathbb{B}}| = |\mathbb{B}| + 1$, *every embedding* $f \colon \mathbb{B} \to \mathbb{A}$ *extends to an embedding* $\hat{f} \colon \widehat{\mathbb{B}} \to \mathbb{A}$.

The property $(*)$ implies that every structure $\mathbb{B} \in \mathcal{C}$ embeds into $\mathbb{A}$, as we can pick a sequence of structures $\mathbb{B}_0 \subseteq \mathbb{B}_1 \subseteq \ldots \subseteq \mathbb{B}_n = \mathbb{B}$ where $\mathbb{B}_0$ is empty and each structure is an induced substructure of the next structure, with one element removed. Repeatedly applying the property $(*)$ leads to an embedding of $\mathbb{B}$ into $\mathbb{A}$, proving that every structure in $\mathcal{C}$ embeds into $\mathbb{A}$. The structure $\mathbb{A}$ constructed below will also have the property that every its finite induced substructure belongs to $\mathcal{C}$. By Proposition 7.9, then property $(*)$ implies that $\mathbb{A}$ has quantifier elimination. Also, Lemma 7.11 proves that $\mathbb{A}$ is unique, up to isomorphism.

The structure $\mathbb{A}$ is constructed in stages, as a union of a chain of finite structures belonging to $\mathcal{C}$:

$$\mathbb{A}_0 \subseteq \mathbb{A}_1 \subseteq \mathbb{A}_2 \subseteq \ldots \tag{7.4}$$

The structure $\mathbb{A}_0$ is the empty structure. For $n > 0$, the structure $\mathbb{A}_n \in \mathcal{C}$ is constructed inductively as follows.

Consider all triples $(\mathbb{B}, \widehat{\mathbb{B}}, f)$ where $\mathbb{B}$ and $\widehat{\mathbb{B}}$ is as in $(*)$ and $f \colon \widehat{\mathbb{B}} \to \mathbb{A}_{n-1}$ is an embedding. Up to isomorphism, there are only finitely many such triples $(\mathbb{B}, \widehat{\mathbb{B}}, f)$. Arrange those triples in a finite sequence arbitrarily. Let $\mathbb{A}_n^0 = \mathbb{A}_{n-1}$ and process the triples sequentially. For the ith triple $(\mathbb{B}, \widehat{\mathbb{B}}, f)$, using amalgamation, construct a structure $\mathbb{A}_n^i \in \mathcal{C}$ containing $\mathbb{A}_n^{i-1}$ as an induced substructure, and such that f extends to an embedding of $\widehat{\mathbb{B}}$ into $\mathbb{A}_n^i$. After all triples have been processed, let $\mathbb{A}_n$ be the last constructed structure $\mathbb{A}_n^i$.

This ends the description of the sequence (7.4). By construction, the union $\bigcup_n \mathbb{A}_n$ satisfies $(*)$, finishing the proof. ■

8 Compositionality

In this chapter, we study an important feature of first-order logic, but also of the more powerful monadic second-order logic. *Compositionality* essentially says that the type of the disjoint union $\mathbb{A} \uplus \mathbb{B}$ of two structures $\mathbb{A}$ and $\mathbb{B}$ depends only on the types of $\mathbb{A}$ and of $\mathbb{B}$. There are two equivalent formalizations of the notion "depends only on":

1. for any structures $\mathbb{A}_1, \mathbb{A}_2, \mathbb{B}_1, \mathbb{B}_2$, the conjunction of the first two equivalences below implies the third:

$$\frac{\begin{array}{ccc} \mathbb{A}_1 & \equiv^r & \mathbb{A}_2 \\ \mathbb{B}_1 & \equiv^r & \mathbb{B}_2 \end{array}}{\mathbb{A}_1 \uplus \mathbb{B}_1 \equiv^r \mathbb{A}_2 \uplus \mathbb{B}_2}$$

2. there is a function $F_r \colon \Delta_r \times \Delta_r \to \Delta_r$, where Δ_r is the set of types of quantifier rank r, such that $\mathrm{tp}^r(\mathbb{A} \uplus \mathbb{B}) = F(\mathrm{tp}^r(\mathbb{A}), \mathrm{tp}^r(\mathbb{B}))$.

It is easy to check that the two conditions above are equivalent.

We say that $\mathrm{tp}^r(\mathbb{A} \uplus \mathbb{B})$ *effectively* depends on $\mathrm{tp}^r(\mathbb{A})$ and on $\mathrm{tp}^r(\mathbb{B})$ if the function $F_r \colon \Delta_r \times \Delta_r \to \Delta_r$ as in the second condition above is uniformly computable in r: there is an algorithm which inputs $r \in \mathbb{N}$ and two types $\tau, \sigma \in \Delta_r$ (represented as trees in $\mathcal{P}(\Delta(r))$) and outputs $F_r(\tau, \sigma)$ (if σ or τ is not satisfiable, the result may be arbitrary).

Theorem 8.1. *Fix $r \in \mathbb{N}$ and a relational signature Σ. For any two structures $\mathbb{A}$ and $\mathbb{B}$ the type $\mathrm{tp}^r(\mathbb{A} \uplus \mathbb{B})$ effectively depends only on the types $\mathrm{tp}^r(\mathbb{A})$ and $\mathrm{tp}^r(\mathbb{B})$.*

Proof. We first exhibit the dependence, and then argue that it is in fact effective.

Let $\mathbb{A}_1, \mathbb{A}_2, \mathbb{B}_1, \mathbb{B}_2$ be structures such that $\mathbb{A}_1 \equiv^r \mathbb{A}_2$ and $\mathbb{B}_1 \equiv^r \mathbb{B}_2$. We prove that $\mathbb{A}_1 \uplus \mathbb{B}_1 \equiv^r \mathbb{A}_2 \uplus \mathbb{B}_2$.

By assumption, duplicator has a winning strategy in the r-round game on $\mathbb{A}_1$ and $\mathbb{A}_2$. She also has a winning strategy in the r-round game on $\mathbb{A}_2$ and $\mathbb{B}_2$. Now, duplicator can combine these strategies to a winning strategy in the game on $\mathbb{A}_1 \uplus \mathbb{B}_1$ and $\mathbb{A}_2 \uplus \mathbb{B}_2$. See Fig. 8.1. This proves that $\mathrm{tp}^r(\mathbb{A} \uplus \mathbb{B})$ depends only on $\mathrm{tp}^r(\mathbb{A})$ and $\mathrm{tp}^r(\mathbb{B})$.

We now briefly argue that this dependency is effective. To this end, we show that $\mathrm{tree}^r(\mathbb{A} \uplus \mathbb{B})$ can be computed given $\mathrm{tree}^r(\mathbb{A})$ and $\mathrm{tree}^r(\mathbb{B})$. Recall that the trees represent all possible sequences of moves of either player in the considered structure of length at most r, and each such sequence is labelled by the substructure induced by the pebbled positions. A sequences of moves in $\mathrm{tree}^r(\mathbb{A} \uplus \mathbb{B})$ is an interleaving of moves in $\mathbb{A}$ and in $\mathbb{B}$, and thus yields two sequences of moves, in $\mathbb{A}$ and in $\mathbb{B}$. The label of such a sequence is uniquely (and effectively) determined by the labels of the two subsequences. This allows to effectively combine the trees $\mathrm{tree}^r(\mathbb{A})$ and $\mathrm{tree}^r(\mathbb{B})$ into $\mathrm{tree}^r(\mathbb{A} \uplus \mathbb{B})$. ∎

Corollary 8.2. *Let φ be a sentence. There is $k \in \mathbb{N}$ and sentences $\alpha_1, \beta_1, \ldots, \alpha_k, \beta_k$, effectively computable from φ, such that for any two structures $\mathbb{A}$ and $\mathbb{B}$,*

$$\mathbb{A} \uplus \mathbb{B} \models \varphi \qquad \Leftrightarrow \qquad \mathbb{A} \models \alpha_i \quad \textit{and} \quad \mathbb{B} \models \beta_i \quad \textit{for some } i \in \{1, \ldots, k\}.$$

Proof. Let Δ_r be the (finite) set of all types of quantifier rank r with no free variables. By Theorem 8, there is a function $F_r \colon \Delta_r \times \Delta_r \to \Delta_r$ such that $\mathrm{tp}^r(\mathbb{A} \uplus$

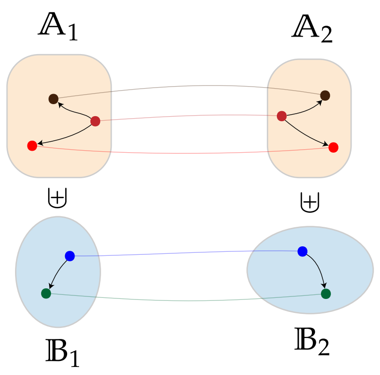


Figure 8.1: The Ehrenfeucht-Fraïssé game on $\mathbb{A}_1 \uplus \mathbb{B}_1$ and $\mathbb{A}_2 \uplus \mathbb{B}_2$. Whenever spoiler plays on $\mathbb{A}_1$ or $\mathbb{A}_2$, duplicator replies according to a strategy in the smaller game on $\mathbb{A}_1$ and $\mathbb{A}_2$. Whenever spoiler plays on $\mathbb{B}_1$ or $\mathbb{B}_2$, duplicator replies according to a strategy in the smaller game on $\mathbb{B}_1$ and $\mathbb{B}_2$. If duplicator can win in the two smaller games, she wins in the combined game. The reason is that the substructure induced by the pebbles in $\mathbb{A}_1 \uplus \mathbb{B}_1$ is the disjoint union of the substructure induced by the pebbles in $\mathbb{A}_1$ and the substructure induced by the pebbles in $\mathbb{B}_1$, and similarly for $\mathbb{A}_2$ and $\mathbb{B}_2$. Hence, the partial isomorphism between $\mathbb{A}_1$ and $\mathbb{A}_2$ induced by the pebbles therein, and the partial isomorphism between $\mathbb{B}_1$ and $\mathbb{B}_2$ induced by the remaining pebbles, together yield a partial isomorphism between $\mathbb{A}_1 \uplus \mathbb{B}_1$ and $\mathbb{A}_2 \uplus \mathbb{B}_2$ between the pebble positions.

$\mathbb{B}) = F_r(\mathrm{tp}^r(\mathbb{A}), \mathrm{tp}^r(\mathbb{B}))$. Then we have the following equivalences:

$$\begin{aligned}
\mathbb{A} \uplus \mathbb{B} \models \varphi &\leftrightarrow \\
\varphi \in \mathrm{tp}^r(\mathbb{A} \uplus \mathbb{B}) &\leftrightarrow \\
\varphi \in F_r(\mathrm{tp}^r(\mathbb{A}), \mathrm{tp}^r(\mathbb{B})) &\leftrightarrow \\
\bigvee_{\substack{\tau,\sigma \in \Delta_r \\ \varphi \in F_r(\tau,\sigma)}} (\mathrm{tp}^r(\mathbb{A}) = \tau) \wedge (\mathrm{tp}^r(\mathbb{B}) = \sigma) &\leftrightarrow \bigvee_{\substack{\tau,\sigma \in \Delta_r \\ \varphi \in F_r(\tau,\sigma)}} (\mathbb{A} \models \widehat{\tau}) \wedge (\mathbb{B} \models \widehat{\sigma}),
\end{aligned}$$

where $\widehat{\tau}$ denotes a sentence of quantifier rank r characterizing the type τ, as given by Corollary 5.14. As Δ_r is finite, this yields the conclusion. Effectivity follows from the uniform computability of the functions $F_r \colon \Delta_r \times \Delta_r \to \Delta_r$. ■

Theorem can be proved more generally for the logic MSO. Let $\mathrm{tp}^q_{\mathrm{MSO}}(\mathbb{A})$ denote the set of MSO sentences of quantifier rank q that hold in $\mathbb{A}$.

Theorem 8.3. *Fix $r \in \mathbb{N}$ and a relational signature Σ. For any two structures $\mathbb{A}$ and $\mathbb{B}$ the type $\mathrm{tp}^r_{\mathrm{MSO}}(\mathbb{A} \uplus \mathbb{B})$ effectively depends only on the types $\mathrm{tp}^r_{\mathrm{MSO}}(\mathbb{A})$ and $\mathrm{tp}^r_{\mathrm{MSO}}(\mathbb{B})$.*

We leave the proof as Exercise 8.6.

Exercise 8.4. Deduce Theorem 8 from Corollary 8.2.

Exercise 8.5. Provide a purely syntactic proof of Corollary 8.2, by extending the logic by two types of quantifiers, $\exists x \in A$ and $\exists x \in B$, where in a structure of the form $\mathbb{A} \uplus \mathbb{B}$, the first quantifier only ranges over the elements of $\mathbb{A}$, while the second quantifier only ranges over the elements of $\mathbb{B}$. Then replace each quantifier $\exists x.\psi$ in φ by a disjuction $\exists x \in A.\psi \vee \exists x \in B.\psi$.

Exercise 8.6. Prove Theorem 8.3, using a variant of Ehrenfeucht-Fraïssé games for MSO (see Exercise 5.22).

Model checking and satisfiablity of MSO over trees. As an application of compositionality of MSO, we prove that the problem of deciding whether a given MSO formula is satisfiable in some (labeled) tree is decidable. We also show that every MSO formula φ can be evaluated on a given input (labeled) tree T in time linear in the size of T.

A signature of *labeled forests* is a signature consisting of a binary relation symbol *parent* and of finitely many unary relation symbols, called the *labels*, and a distinguished unary relation *root*. Fix such a signature Σ in what follows. A *rooted labeled forest* is a Σ-structure F such that every root v of F has no parent (that is, $\neg\exists x, y.root(x) \wedge parent(y, x)$), every non-root element v of F has exactly one parent and moreover there is some $d \in \mathbb{N}$ and a finite sequence $v_0, v_1, v_2, \ldots, v_d$ such that $v = v_0$, v_{i+1} is the parent of v_i for $0 \leqslant i < d$, and such that v_d is a root. Elements of a forest are called *nodes*. A *tree* is a forest with exactly one root. Removing the root r from a tree T results in a forest T', defined as the substructure of T induced by $V(T) - \{r\}$, with all the children of r in T marked as roots.

Lemma 8.7. *Fix $q \in \mathbb{N}$. Let T be a tree and let T' be the forest obtained from T by removing the root r. Then $\mathrm{tp}^q_{\mathrm{MSO}}(T)$ depends only on $\mathrm{tp}^q_{\mathrm{MSO}}(T')$ and on $\mathrm{atp}_T(r)$.*

Theorem 8.8. *Let φ be a sentence of monadic second-order logic. There is an algorithm which given a finite rooted labeled tree T decides in time $\mathcal{O}_\varphi(|T|)$ whether T satisfies φ.*

Proof. Fix a number $q \in \mathbb{N}$, and let

$$\Delta_q = \left\{ \mathrm{tp}^q_{\mathrm{MSO}}(\mathbb{A}) \mid \mathbb{A} \text{ is a } \Sigma\text{-structure} \right\}.$$

Then Δ_q is finite, by Corollary 5.13. Let $f_\uplus \colon \Delta_q \times \Delta_q \to \Delta_q$ be such that for every two Σ-structures $\mathbb{A}$ and $\mathbb{B}$, $\mathrm{tp}^q_{\mathrm{MSO}}(\mathbb{A} \uplus \mathbb{B}) = f_\uplus(\mathrm{tp}^q_{\mathrm{MSO}}(\mathbb{A}), \mathrm{tp}^q_{\mathrm{MSO}}(\mathbb{B}))$. Such a function $f_\uplus$ exists by Theorem 8.3. For every atomic type $\tau(x)$ with one free variable let $f_\tau \colon \Delta_q \to \Delta_q$ be such that for every finite rooted labeled tree T whose root r satisfies $\mathrm{atp}_F(r) = \tau$, $\mathrm{tp}^q_{\mathrm{MSO}}(T) = f_\tau(\mathrm{tp}^q_{\mathrm{MSO}}(F))$, where F is the forest obtained from T by removing the root r. Such a function exists by Lemma 8.7.

Every forest F is a disjoint union of trees, and every tree T is obtained from some forest by prepending a root. Thus, we may compute $\mathrm{tp}^q_{\mathrm{MSO}}(F)$ by induction on $|F|$:

- if $|F| = 0$ then $\mathrm{tp}^q_{\mathrm{MSO}}(F) = \mathrm{tp}^q_{\mathrm{MSO}}(\emptyset)$, where $\emptyset$ is the empty Σ-structure,
- if F is a tree then $\mathrm{tp}^q_{\mathrm{MSO}}(F) = f_\tau(\mathrm{tp}^q_{\mathrm{MSO}}(F'))$, where F' is obtained from F by removing its root,
- otherwise, F is a disjoint union $F_1 \uplus F_2$ of two nonempty forests, and $\mathrm{tp}^q_{\mathrm{MSO}}(F) = f_\uplus(\mathrm{tp}^q_{\mathrm{MSO}}(F_1), \mathrm{tp}^q_{\mathrm{MSO}}(F_2))$.

Note that each of the above cases corresponds to applying a function to some elements of the set Δ_q (in the first case, a constant, in the second case, a unary function, and in the third case, a binary function). As Δ_q is a fixed, finite set (depending on φ, which is fixed), each such function can be computed in constant time. Therefore, this recursion gives a linear time algorithm for computing $\tau := \mathrm{tp}^q_{\mathrm{MSO}}(F) \in \Delta_q$. Having computed τ, output yes or no depending on whether or not $\varphi \in \tau$ (this bit is hard-coded into the algorithm, for every $\tau \in \Delta_q$). ■

A few words regarding the effectiveness of the above algorithm are in place. We have obtained a *non-uniform* algorithm for model-checking: for every MSO formula φ, we provided a separate algorithm for model-checking φ on forests. In the uniform version, φ is not fixed, and is given on input. It is not difficult to prove the following, uniform strengthening of Theorem :

Theorem 8.9. *There is an algorithm which, given an MSO-sentence φ of quantifier rank q over a signature Σ of rooted labeled trees, and a finite rooted labeled tree T, decides whether T satisfies φ in time $f(q)(|T|)$, for some computable function $f\colon \mathbb{N} \to \mathbb{N}$.*

Proof. By revisiting the proof of Theorem 8, and verifying that each step is indeed effective. This analysis is based on the explicit representation of types as trees (see the proof of Theorem 5.8), the fact that we can effectively rewrite a given sentence φ into a disjunction of such trees (see Lemma 5.17), and the fact that the functions $f_\uplus$ and f_τ are computable, once the types are represented as trees. This follows from the explicit constructions in the proofs of Theorem 8.3 and Lemma 8.7. We omit the details. ■

Theorem 8.10. *It is decidable whether a given sentence φ of monadic second-order logic is satisfied in some finite rooted labeled tree.*

Proof. Let $S_q \subseteq \Delta_q$ be the set of all MSO-types of quantifier rank q of finite labeled rooted forests:

$$S_q = \left\{ \mathrm{tp}^q_{\mathrm{MSO}}(T) \mid T \text{ is a finite labeled rooted forest} \right\}.$$

Then $S_q \subseteq \Delta_q$ is the smallest set with the following properties:

- S_q contains the MSO-type of quantifier rank q of the empty forest, $\mathrm{tp}^q_{\mathrm{MSO}}(\emptyset)$,
- S_q is closed under $f_\uplus\colon \Delta_q \times \Delta_q \to \Delta_q$ and under $f_\tau\colon \Delta_q \to \Delta_q$, for every atomic type $\tau(x)$.

By the discussion in the proof of Theorem 8.9, those functions can be effectively represented and computed, given a number q. In particular, the set S_q is computable by a fixpoint algorithm, given q. Next, rewrite φ as a disjunction of types $\tau_1 \vee \cdots \vee \tau_k$ (represented as trees, some of which may be unsatisfiable, see Lemma 5.17). Then φ is satisfiable in a finite forest if and only if $\tau_i \in S_q$ for some $i = 1, \ldots, k$. ■

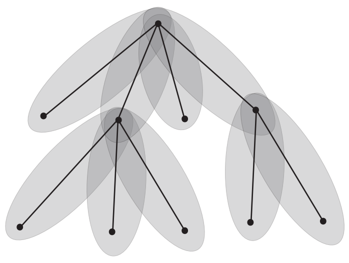
8.1 Treewidth

A *tree decomposition* of a graph G is a (unrooted, unranked) tree T such that the following conditions hold (cf. Fig. 8.2):

1. each node of T is a set of vertices of G; the nodes of T are also called *bags* of the decomposition;
2. for every vertex v of G, the set of nodes of T containing v is a nonempty connected subtree of T;
3. for every edge uv of G there exists $b \in V(T)$ such that $u, v \in b$.

The *width* of a tree decomposition T is $\max_{b \in V(T)} |b| - 1$ and the *treewidth* of a graph G is the minimum width of a tree decomposition of G.

Example 8.11. Every tree has a tree decomposition of width 1, as depicted below:



Here, we only depict the tree G and highlight the bags of the decomposition, each containing 2 vertices of G. The tree decomposition T is just the tree G

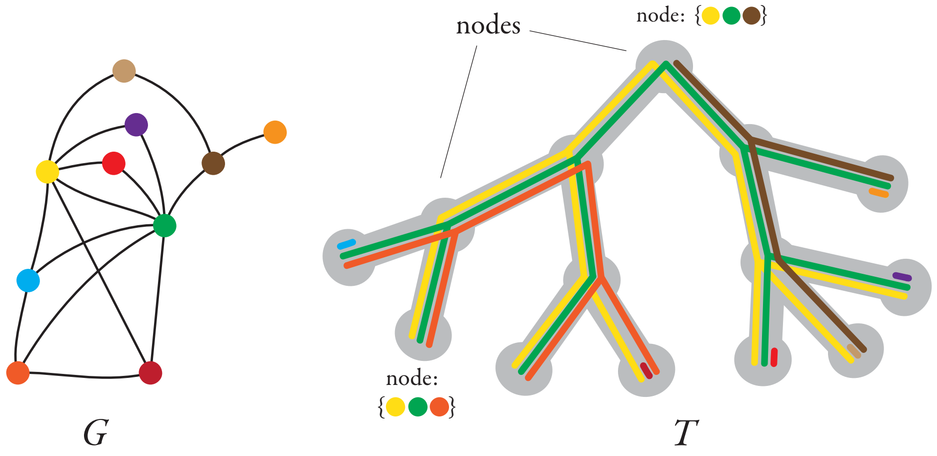
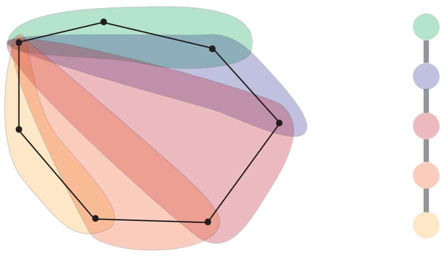


Figure 8.2: A graph G and its tree decomposition T of width $k = 2$. A node $t \in T$ is a set of at most $k+1$ vertices of G. The set of nodes of T whose bag contains a given vertex $v \in V(G)$ forms a connected subtree of T, marked with the matching color. For every edge $uv \in E(G)$ there is some node $t \in T$ with $u, v \in t$.

itself, where the bag of a node v consists of v and its parent (if it exists) with respect to an arbitrarily chosen root. Every forest also has a tree decomposition of width 1, as a forest is a subgraph of a tree, and in general, if $H \subseteq G$ then $\text{treewidth}(H) \leqslant \text{treewidth}(G)$.

A cycle C has treewidth at most 2, as depicted below:



Exercise 8.12. Show that a cycle of length 3 or more has no tree decomposition of width 1. It follows that the graphs of treewidth 1 are precisely all forests. More generally, show that the treewidth of the t-clique is $t - 1$.

Game characterisation. Treewidth can be seen as a game on a graph, although the proof of the equivalence is far from trivial. The players of the game are called *cops* and the *robber*. The game is parameterized by an integer k, the number of cops. The graph G is again the arena of the game. At every step of the game, the robber occupies a vertex $r \in V(G)$ and cops occupy a set $C \subseteq V(G)$ of size at most k of vertices (every cop occupies a vertex). The game ends immediately with a win of the cops if some cop is at the same vertex as the robber, that is, if $r \in C$.

Initially, first the cop player places k cops at vertices of the graph (chooses the initial set C) and then the robber player places the robber at one vertex of the graph (chooses the initial vertex r). At every round:

- the cop player declares, for every cop, whether the cop stays at the current vertex or moves to another vertex (in other words, declares the new set $C' \subseteq V(G)$ where the cops will be placed at the end of the round);
- the cops that are supposed to move to another vertex leave their vertices (i.e., now only the vertices of $C \cap C'$ are occupied by cops);
- the robber moves to any vertex r' by traversing edges and vertices that are not occupied by cops (i.e., the robber chooses the next vertex r' as any vertex of the connected component of $G - (C \cap C')$ that contains the current vertex r);
- the cops that are supposed to move arrive at their target vertices (i.e., now cops occupy the vertex set C').

If the robber is able to indefinitely avoid being caught, then the robber player wins.

It can be proved that the minimum number of cops needed to catch the robber is exactly one more than the treewidth of G. The easier direction of this statement is that any (in particular, an optimum-width) treewidth decomposition can be interpreted as a strategy for the cops that move bag-by-bag in the direction of the robber. The strategy for the robber in case of smaller-than-necessary number of cops is given by the notion of a *bramble*, which is beyond the scope of this book.

Algebraic characterisation. The following yields an operational description of graphs of bounded treewidth. A graph with *k ports* is a graph G together with a labeling of its vertices with numbers from $1, \ldots, k$, so that every vertex is labeled by at most one number, and every number appears as a label of at most one vertex.

Let G, H be two graphs with k ports. Write $G \cup_k H$ to denote the graph obtained by first taking the disjoint union of G and H, and then identifying pairs of vertices which have the same label in G and H. When identifying a pair of vertices u and v we obtain a vertex $\{u, v\}$ which is adjacent both to the original neighbors of u and to the neighbors of v. The obtained graph $G \cup_k H$ is a graph with k ports, where the labels are inherited from the labels in G and in H.

Say that a graph G with k ports is *k-constructible* if G can be constructed by recursively using the following operations:

1. Construct a graph G with a single vertex v, which is labeled i, for some $i \in \{1, \ldots, k\}$,

2. Given a graph G with k ports, erase a label $i \in \{1, \ldots, k\}$,

3. Given two graphs G, H with k ports, produce the graph $G \cup_k H$ with k ports.

These operations can be seen as k constants, one for each $i = 1, \ldots, k$ (defining a graph with a single bertex labeled i), a unary operation, for each $i = 1, \ldots, k$ (forgetting the label i), and a binary operation $\bigcup_k$. A *treewidth-k term* is a term producing a graph G with k ports using those operations.

Proposition 8.13. *Fix $k \in \mathbb{N}$. A graph G has treewidth at most $k-1$ if and only if it is $k+1$-constructible. Moreover, there is a linear-time algorithm that, given a tree decomposition of G of width k, outputs a treewidth-k term producing G, and vice-versa.*

Exercise 8.14. Prove Proposition 8.13.

Complexity. It is NP-complete to decide whether a given graph G has treewidth bounded by a given number k. However, if k is considered fixed, then this can be decided in time linear in the size of G:

Theorem 8.15. *There is a computable function $f\colon \mathbb{N} \to \mathbb{N}$ and an algorithm which inputs a graph G and a number k and in time $f(k) \cdot |G|$ either computes a tree decomposition of G of width at most k, or declares that no such decomposition exists.*

MSO satisfiability and model checking. The following lemma is an generalizes Lemma 8.7 to the case of graphs of operations that are pertinent to bounded treewidth, as described in Proposition 8.13.

For two graphs G, H with not necessarily disjoint domains let $G \cup H$ denote their union.

Lemma 8.16. *Fix $q \in \mathbb{N}$. Let G and H be two graphs with $V(G) \cap V(H) = \{s_1, \dots, s_k\}$. Then $\mathrm{tp}^q_{\mathrm{MSO}}(G \cup H, \bar{s})$ depends only on $\mathrm{tp}^q_{\mathrm{MSO}}(G, \bar{s})$ and $\mathrm{tp}^q_{\mathrm{MSO}}(H, \bar{s})$.*

Proof. We leave the proof as Exercise 8.17. ∎

Exercise 8.17. Prove Lemma 8.16.

Let $\mathcal{T}_k$ denote the class of finite graphs of treewidth at most k. We consider the satisfiability problem for monadic second-order logic over $\mathcal{T}_k$, where we are given a sentence φ of MSO and a number $k \in \mathbb{N}$, and the task is to decide whether φ is satisfied in some graph $G \in \mathcal{T}_k$.

Theorem 8.18. *Satisfiability of monadic second-order logic is decidable over $\mathcal{T}_k$.*

Proof. Analogous to Theorem 8.10, using Lemma 8.16, and the recursive description of graphs of treewidth k given by Proposition 8.13. ∎

Proposition 8.19. *There is a computable function $f\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ and an algorithm which inputs a sentence φ of monadic second-order logic and a graph G together with its tree decomposition T of width k, and decides if G satisfies φ in time $f(k, |\varphi|)(|G|)$.*

Proof. Analogous to Theorem 8.9, using Proposition 8.13. ∎

Together with Theorem 8.15, this yields the following result:

Theorem 8.20 (Courcelle)**.** *There is a computable function $f\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ and an algorithm which inputs a sentence φ of monadic second-order logic and a graph G of treewidth at most k, and decides if G satisfies φ in time $f(k, |\varphi|)(|G|)$.*

8.2 Cliquewidth

Graphs of bounded treewidth are sparse: a graph with n vertices of treewidth k has at most $\mathcal{O}_k(n)$ edges. In particular, the class of all cliques does not have bounded treewidth; yet those graphs are simply complements of edgeless graphs. The aim now is to generalize the notion of treewidth to yield a notion which is invariant under edge complementation.

Fix a finite set of colors C. A *C-colored graph* is a graph G in which each vertex has a color from C assigned to it. We define the following operations on C-colored graphs:

constant for each $i \in C$, the graph with one vertex colored i is a C-colored graph, denoted c_i;

recolor for every function $f\colon C \to C$ there is an operation $\mathit{recolor}_f$ which inputs a C-colored graph G and outputs the C-colored graph obtained from G by replacing the color i of a vertex by the color $f(i)$;

join for every set $M \subseteq \{\{i,j\} \mid i,j \in C\}$ there is an operation join_M which inputs two C-colored graphs G and H, and outputs the disjoint union of G and H (with each vertex retaining its color) extended by the edges connecting each vertex v of color i of G with each vertex of color j of H if and only if $\{i,j\} \in M$.

A *clique decomposition* with colors C is a term t without free variables built out of the above operations. It therefore defines a C-colored graph, which we denote $[t]$. Formally, $[t]$ is the graph whose vertices are the leaves of t, and is defined inductively, as expected. The *width* of a clique decomposition is $|C|$. A graph G has *cliquewidth* k if there is a clique decomposition t of width k which produces a graph isomorphic to G together with some coloring.

Example 8.21. Every clique has cliquewidth 1. Every biclique has cliquewidth 2. Every cograph has cliquewidth 2. Every graph of treewidth k has cliquewidth $\mathcal{O}(2^k)$.

The term t can be viewed as a ranked tree whose leaves are labeled with c_i, for $i \in C$, and inner nodes have either rank 1 and have label $recolor_f$, for $f\colon C \to C$, or have rank 2 and have label $join_M$, for $M \subseteq \{\{i,j\} \mid i,j \in C\}$. Observe that the order of children does not matter in the tree, since the join operation is commutative.

We will use the following result, which implies that for every class of bounded clique-width there is an algorithm that computes a clique decomposition of bounded width, in cubic time.

Theorem 8.22 (Courcelle, Oum). *There is a computable function $f\colon \mathbb{N} \to \mathbb{N}$ and an algorithm which inputs a graph G and a number k and in time $f(k) \cdot |G|^3$ either computes a clique decomposition of G of width at most k, or declares that no such decomposition exists.*

A clique decomposition with colors C can be seen either as a ranked tree over a ranked alphabet

$$A_C = \{c_i \mid i \in C\} \cup \left\{recolor_f \mid f\colon C \to C\right\} \cup \{join_M \mid M \subseteq \{\{i,j\} \mid i,j \in C\}\},$$

or as a relational structure equipped with the child relation, and unary predicates corresponding to the labels in A_C.

Corollary 8.23. *Satisfiability of monadic second-order logic is decidable over $\mathcal{C}_k$.*

Corollary 8.24. *There is an algorithm which inputs a sentence φ of monadic second-order logic and a graph G together with its clique decomposition t of width k, and decides if G satisfies φ in time $\mathcal{O}_{k,\varphi}(|t|)$.*

Together with Theorem 8.22, this yields:

Theorem 8.25. *There is an algorithm which inputs a sentence φ of monadic second-order logic and a graph G of cliquewidth at most k, and decides if G satisfies φ in time $\mathcal{O}_{k,\varphi}(|G|^3)$.*

We finish with the following, logical characterisation of classes with bounded clique-width. A rooted tree T vertex-colored with k colors is modelled as a structure with a binary parent relation E and k unary relations, one per each color. Let $\varphi(x,y)$ be a formula in this language, and T be such a rooted tree. By $\varphi(T)$ denote the graph G whose vertices are the leaves of T, and where two distinct leaves u,v of T are adjacent in G if and only if $T \models \varphi(u,v) \vee \varphi(v,u)$.

Theorem 8.26. *Let $\mathcal{C}$ be a class of graphs. The following conditions are equivalent:*

- *$\mathcal{C}$ has bounded clique-width,*
- *there is a number k and an MSO-formula $\varphi(x,y)$ in the language of rooted trees vertex-colored with k colors, such that every $G \in \mathcal{C}$ is of the form $\varphi(T)$ for some rooted tree T, vertex-colored with k colors.*

See Appendix B.1 for a proof. We remark that the same holds for first-order formulas instead of MSO-formulas, but now a rooted tree T needs to be viewed as a structure over a richer language, containing additionally the ancestor relation $\leqslant$, which may be used in the first-order formula $\varphi(x,y)$.

9
Locality

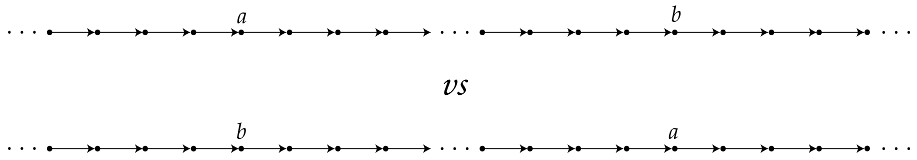
In this chapter we study a feature that is specific to first-order logic (and some other logics), namely locality.

While for every fixed number d we can write down a first-order formula $\mathrm{dist}(x,y) \leqslant d$ stating that x and y are at distance at most d, to express that two vertices lie in the same component of a graph we intuitively require an unbounded number of quantifiers. This intuition that first-order logic can express only *local* properties turns out to be true and can be formalized in several ways, for example by the locality theorems of Hanf and Gaifman. In this chapter we are going to study Gaifman's Locality Theorem, which forms the basis for an efficient model-checking algorithm for first-order formulas on graphs of bounded degree, and other graph classes. The locality results imply, in particular, that whether or not a tuple $\bar{v} \in G^{\bar{x}}$ of vertices of a graph G satisfies a fixed formula $\varphi(\bar{x})$, depends only on a neighborhood of $\bar{v}$ in G of radius depending on φ only.

Example 9.1. Consider a directed path P_n with n vertices. We would like to argue that there is no formula $\varphi(x,y)$ expressing that x lies before y on the path. More formally, there is no $\varphi(x,y)$ such that for every $n \in \mathbb{N}$ and elements $a, b \in P_n$, $P_n \models \varphi(a,b)$ if and only if b is reachable from a by a directed path.

This can be proved using an Ehrenfeucht-Fraïssé argument, however, in this chapter we would like to obtain a more general understanding of this type of phenomena.

The intuition that will be made formal is that a formula $\varphi(a, b)$ can only see a local neighborhood around a and b, and if a and b are far enough (with respect to the quantifier rank of φ), then the formula won't be able to tell apart the two situations depicted below:



Gaifman graph

First we define the notion of a neighboorhood in an arbitrary relational structure.

Fix a relational signature Σ. The *Gaifman graph* of a Σ-structure $\mathbb{A}$ is the (simple, undirected) graph whose vertices are the elements of $\mathbb{A}$, and in which two distinct elements a, b are adjacent if there is a relation $R \in \Sigma$ and tuple $\bar{a} \in R_{\mathbb{A}}$ which contains both a and b. For a structure $\mathbb{A}$ and tuple $\bar{a}b$ of its elements, write $\text{dist}(\bar{a}, b) \leqslant r$ to denote that b is within distance at most r from one of the vertices in $\bar{a}$ in the Gaifman graph of $\mathbb{A}$. Denote

$$N_r(\bar{a}) = \{b \in \mathbb{A} \mid \text{dist}(\bar{a}, b) \leqslant r\},$$

and let $\mathbb{A}[N_r(\bar{a})]$ denote the substructure of $\mathbb{A}$ induced by $N_r(\bar{a})$.

Local types

One way of formalizing the intuition presented above is by using the following notion.

Definition 1. Let $\mathbb{A}$ and $\mathbb{B}$ be structures and $\bar{a} \in \mathbb{A}^k$ and $\bar{b} \in \mathbb{B}^k$ be tuples of the same length. The *local r-round EF game* on $(\mathbb{A}, \bar{a})$ and $(\mathbb{B}, \bar{b})$ is defined the same way as the usual EF game, starting with pebbles on $\bar{a}$ and $\bar{b}$, but in the ith round of the game, Spoiler can play only within distance (in the Gaifman graph) at most 2^{r-i} from the vertices pebbled so far. In each move, Duplicator must guarantee the atomic type of the tuple $(a_1, \dots, a_k)$ of vertices pebbled in $\mathbb{A}$ and the tuple $(b_1, \dots, b_k)$ of vertices $\mathbb{B}$ pebbled in $\mathbb{B}$ are equal, and moreover, for every $1 \leqslant i, j \leqslant k$, a_i and a_j are adjacent in the Gaifman graph of $\mathbb{A}$ if and only if b_i and b_j are adjacent in the Gaifman graph of $\mathbb{B}$.

A more formal definition is as follows. Define the relation $\bar{a} \simeq^{\text{loc}}_q \bar{b}$, where $\bar{a}$ and $\bar{b}$ are tuples of elements of the same length belonging to two structures $\mathbb{A}$ and $\mathbb{B}$, respectively, inductively as follows:

- For $r = 0$, write $\bar{a} \simeq^{\text{loc}}_r \bar{b}$ if $\text{atp}_{\mathbb{A}}(\bar{a}) = \text{atp}_{\mathbb{B}}(\bar{b})$ and $\text{atp}_{G_{\mathbb{A}}}(\bar{a}) = \text{atp}_{G_{\mathbb{B}}}(\bar{b})$, where $G_{\mathbb{A}}$ and $G_{\mathbb{B}}$ are the Gaifman graphs of $\mathbb{A}$ and $\mathbb{B}$, respectively.
- For $r \geqslant 1$, write $\bar{a} \simeq^{\text{loc}}_r \bar{b}$ if for every vertex v within distance at most 2^{q-1} from either of the vertices in $\bar{a}$ there is a vertex w such that $\bar{a}v \simeq^{\text{loc}}_{r-1} \bar{b}w$ and conversely, for every vertex w within distance at most 2^r from either of the vertices in $\bar{b}$ there is a vertex v such that $\bar{a}v \simeq^{\text{loc}}_{r-1} \bar{b}w$.

The $\simeq^{\text{loc}}_r$-equivalence class of a tuple $\bar{a}$ is called its *local type* of rank r, and is denoted $\text{ltp}^r(\bar{a})$.

Note that in the local game, we impose a restriction on the moves of Spoiler – he needs to place a pebble y on a vertex w within distance at most 2^{r-i} of a vertex w already occupied by a pebble x – but Duplicator is unconstrained. However, it turns out that Duplicator needs to place her pebble y on some vertex w' whose distance from the vertex v' with pebble x satisfies $\text{dist}(v', w') = \text{dist}(v, w)$, or otherwise she will lose. This is implied by the following lemma. For $r, k \in \mathbb{N}$ let $[k]_r \in \{0, \dots, r\}$ denote $\min(k, r)$.

Lemma 9.2. *Let* $a_1, a_2 \in \mathbb{A}$ *and* $b_1, b_2 \in \mathbb{B}$ *be such that* $[\text{dist}(a_1, a_2)]_{2^r} \neq [\text{dist}(b_1, b_2)]_{2^r}$. *Then* $(a_1, a_2) \not\simeq^{loc}_r (b_1, b_2)$.

Proof. We proceed by induction on r. The base case follows by definition, since $(a_1, a_2) \simeq^{\text{loc}}_0 (b_1, b_2)$ implies in particular that a_1 is adjacent to a_2 in the Gaifman graph of $\mathbb{A}$ if and only if b_1 is adjacent in b_2 in the Gaifman graph of $\mathbb{B}$.

Suppose the statement holds for r; we prove it for $r + 1$. By symmetry, we may assume that $\text{dist}(a_1, a_2) < \text{dist}(b_1, b_2)$. It follows that $\text{dist}(a_1, a_2) < 2^{r+1}$. Then there is some $a \in \mathbb{A}$ such that $\text{dist}(a, a_1) + \text{dist}(a, a_2) = \text{dist}(a_1, a_2)$ and

$\mathrm{dist}(a, a_1) \leqslant 2^r$ and $\mathrm{dist}(a, a_2) \leqslant 2^r$. Since $\mathrm{dist}(b_1, b_2) > \mathrm{dist}(a_1, a_2)$, there is no $b \in \mathbb{B}$ such that $\mathrm{dist}(b, b_1) \leqslant \mathrm{dist}(a, a_1)$ and $\mathrm{dist}(b, b_2) \leqslant \mathrm{dist}(a, a_2)$. In particular, for every $b \in \mathbb{B}$, either $[\mathrm{dist}(b, b_1)]_{2^r} \neq [\mathrm{dist}(a, a_1)]_{2^r}$, or $[\mathrm{dist}(b, b_2)]_{2^r} \neq [\mathrm{dist}(a, a_2)]_{2^r}$. By inductive assumption, in the first case we have $(a_1, a) \not\simeq_r^{\mathrm{loc}} (b_1, b)$, and in the second case, $(a_2, a) \not\simeq_r^{\mathrm{loc}} (b_2, b)$. In any case, $(a_1, a_2, a) \not\simeq_r^{\mathrm{loc}} (b_1, b_2, b)$. Hence, there is some $a \in \mathbb{A}$ (Spoiler's move) such that for every $b \in \mathbb{B}$ (Duplicator's response) $(a_1, a_2, a) \not\simeq_r^{\mathrm{loc}} (b_1, b_2, b)$ (Duplicator loses in r rounds). By definition, this means that $(a_1, a_2) \not\simeq_{r+1}^{\mathrm{loc}} (b_1, b_2)$, finishing the inductive proof. ∎

Lemma 9.3. *Fix $r, k \in \mathbb{N}$. There are finitely many equivalence classes of the equivalence relation $\simeq_r^{loc}$ on all k-tuples of elements of arbitrary structures.*

Proof. Easy induction on r. ∎

Lemma 9.4. *Let a and b be two elements of the same structure $\mathbb{A}$, such that their distance in the Gaifman graph is larger than 2^q, for some $q \geqslant 0$. If $\mathrm{ltp}^q_{\mathbb{A}}(a) = \mathrm{ltp}^q_{\mathbb{A}}(b)$ then $\mathrm{tp}^q_{\mathbb{A}}(ab) = \mathrm{tp}^q_{\mathbb{A}}(ba)$. In particular, $\mathrm{tp}^q_{\mathbb{A}}(a) = \mathrm{tp}^q_{\mathbb{B}}(b)$.*

Proof. Fix duplicator's strategy in the q-round local game for a and b. We use this strategy to play the usual, global q-round EF game on $(\mathbb{A}, ab)$ and $(\mathbb{A}, ba)$.

Let x and y be the names of the first two pebbles, corresponding to the elements a and b in the first structure, and to the elements b and a in the second structure.

Throughout the game, some pebbles will be marked as *x-local* (close to the initial pebble x), some as *y-local* (close to the initial pebble y) and the rest as *distant*. Initially, x is x-local and y is y-local. The invariant is that after the ith round of the game, we arrive at structures $(\mathbb{A}, \bar{a})$ and $(\mathbb{A}, \bar{a}')$, where $\bar{a}, \bar{a}' \in \mathbb{A}^{\bar{x}}$ and $\bar{x}$ is a collection of pebbles, such that:

- for any x-local pebble x' and y-local pebble y', $\mathrm{dist}(\bar{a}(x'), \bar{a}(y')) > 2^{q-i}$ and $\mathrm{dist}(\bar{a}'(x'), \bar{a}'(y')) > 2^{q-i}$, and
- for any two pebbles x' and y', $[\mathrm{dist}(\bar{a}(x'), \bar{a}(y'))]_{2^{q-i}} = [\mathrm{dist}(\bar{a}'(x'), \bar{a}'(y'))]_{2^{q-i}}$.

Initially (after the 0th round) this holds by assumption.

If Spoiler plays pebble x_i in round i on a vertex w within distance at most 2^{q-i} from a vertex v carrying an x-local pebble z in the first or the second structure, then the pebble x_i is marked as x-local, and Duplicator responds using the local strategy in the other structure, placing her pebble x_i on a vertex w'. By Lemma 9.2, this satisfies the second part of the invariant. The first part of the invariant follows. We proceed similarly if spoiler plays near a vertex with a y-local pebble.

By the invariant, at most of those cases occurs. If neither of those cases occurs, the vertex v pebbled by Spoiler is marked as distant, and Duplicator responds by putting her pebble at the same vertex v.

It is easy to check that this preserves the invariant. In particular, Duplicator wins. ∎

Exercise 9.5. Show that the distance assumption in Lemma 9.4 cannot be dropped, already for $q = 1$.

Lemma 9.6. *For every $q, k, d \in \mathbb{N}$ there is a number $N \in \mathbb{N}$ and an algorithm that given a graph G with maximum degree d together with a k-tuple $\bar{v} \in V(G)^k$ of vertices, computes in time $\mathcal{O}_{q,k,d}(|G|)$ a set $R \subseteq V(G)$ of size at most N such that for every $a \in V(G)$ there is some $b \in R$ with $\mathrm{tp}^q(G, \bar{v}, a) = \mathrm{tp}^q(G, \bar{v}, b)$.*

Exercise 9.7. Prove Lemma 9.6, by generalizing Lemma 9.4 to graphs with k constants $v_1, \dots, v_k$.

Lemma 9.8. *Fix $q, k, d \geqslant 0$ with $k \leqslant q$. There is an algorithm that given a graph G with maximum degree d together with a k-tuple $\bar{v} \in V(G)^k$ of vertices and a formula $\varphi(x_1, \dots, x_k)$ of quantifier rank q, decides in time $\mathcal{O}_{q,k,d}(|G|)$ whether $G \models \varphi(v_1, \dots, v_k)$.*

Proof. By induction on q. The base case $q = 0$ is immediate. Assume the statement holds for q, and we prove it for $q + 1$. Given a graph G and vertices $v_1, \dots, v_k$, compute the set R as in Lemma 9.6, in time linear in $|G|$. Let φ be a formula of quantifier rank $q + 1$. Then φ is a boolean combination of formulas of the form $\exists y.\psi(x_1,\ldots,x_k,y)$, where ψ has quantifier-rank q. It is enough to consider a single formula of this form.

Then the following conditions are equivalent:

- $G \models \exists y.\psi(v_1,\ldots,v_k,y)$,
- there is some $b \in R$ such that $G \models \psi(v_1,\ldots,v_k,b)$.

As R has size $\mathcal{O}_{q,k,d}(1)$ and ψ has quantifier-rank q, the second condition can be verified in time $\mathcal{O}_{q,k,d}(|G|)$, by inductive assumption. ■

Gaifman locality

In this section we prove Gaifman's locality theorem, which informally states that every formula $\varphi(\bar{x})$ is equivalent to a boolean combination of formulas of the following form:

- *local* formulas $\psi(\bar{x})$, which depend only on the local properties around the elements $\bar{x}$,
- *global* sentences γ, which do not depend on $\bar{x}$, but only depend on the entire structure.

Let us formalize first the notion of local properties.

For a non-negative integer r, a formula $\psi(\bar{x})$ is called *r-local* if for all structures $\mathbb{A}$ and all $\bar{a} \in \mathbb{A}^{\bar{x}}$

$$\mathbb{A}[N_r(\bar{a})] \models \psi(\bar{a}) \iff \mathbb{A} \models \psi(\bar{a}).$$

We now describe in greater detail the form of the global sentences γ mentioned above. They will be of a very specific form. A sentence is *basic local with parameters r and s* if it has the form

$$\exists x_1 \ldots \exists x_s \Big(\bigwedge_{1 \leqslant i < j \leqslant s} (\mathrm{dist}(x_i,x_j) > 2r) \wedge \bigwedge_{1 \leqslant i \leqslant s} \psi(x_i) \Big),$$

where $\psi(x)$ is an r-local first-order formula, and $\mathrm{dist}(x,y)$ refers to the distance in the Gaifman graph of the underlying structure.

A basic local sentence hence says that there is an r-scattered set R of size s such that all elements $v \in R$ satisfy the local formula ψ. It is not difficult to see that $\mathrm{dist}(x,y) \leqslant r$ is expressible by a first-order formula $\delta_r(x,y)$ in the signature of $\mathbb{A}$, so a basic local sentence can be in fact written in the vocabulary of $\mathbb{A}$.

We can now state Gaifman's Locality Theorem.

Theorem 9.9. *Every first-order formula $\varphi(\bar{x})$ of quantifier rank k can be effectively translated into an equivalent formula $\psi(\bar{x})$, which is a Boolean combination of r-local formulas and basic local sentences with parameters $r \leqslant 7^k$ and $s \leqslant k + |\bar{x}|$.*

Corollary 9.10. *Every first-order sentence can be translated into an equivalent Boolean combination of basic local sentences.*

As another corollary we get that the satisfaction of a fixed first-order property in a fixed structure by a tuple of elements only depends on its local neighborhood.

Corollary 9.11. *For every first-order formula $\varphi(\bar{x})$ there is a radius $r \in \mathbb{N}$ such that for every structure $\mathbb{A}$ and tuples $\bar{a}, \bar{b} \in \mathbb{A}^{\bar{x}}$, if $\mathbb{A}[N_r(\bar{a})]$ and $\mathbb{A}[N_r(\bar{b})]$ are isomorphic via an isomorphism which maps $\bar{a}$ to $\bar{b}$, then*

$$\mathbb{A} \models \varphi(\bar{a}) \quad \Longleftrightarrow \quad \mathbb{A} \models \varphi(\bar{b}).$$

Example 9.12. We show that the property considered in Example 9.1, that y is reachable by a directed path from x, is not definable by any first-order formula $\varphi(x,y)$.

Consider a formula $\varphi(x,y)$ and let $r \in \mathbb{N}$ be as in Corollary 9.11. Let $\mathbb{A}$ be the directed path with $5r$ vertices and consider any two vertices $a, b \in \mathbb{A}$ whose r-neighborhoods in $\mathbb{A}$ are pairwise disjoint, and do not contain the endpoints of $\mathbb{A}$. Then $\mathbb{A}[N_r(a,b)]$ and $\mathbb{A}[N_r(b,a)]$ are both disjoint unions of directed paths of length $2r$, and are isomorphic via an isomorphism mapping a to b and b to a. Hence, $\mathbb{A} \models \varphi(a,b)$ if and only if $\mathbb{A} \models \varphi(b,a)$, so $\varphi(x,y)$ cannot express that a is before b.

Exercise 9.13. Show that graph planarity is not expressible in first-order logic.

Example 9.14. Consider a set A equipped with a unary predicate U, and the formula $\varphi(x_1, \ldots, x_l)$ expressing that there is some element in U that is different from $x_1, \ldots, x_l$. Then $\varphi(a_1, \ldots, a_l)$ holds for a given tuple $a_1, \ldots, a_l \in A$ if and only if $|U| > |U \cap \{a_1, \ldots, a_l\}|$. Since $|U \cap \{a_1, \ldots, a_l\}| \leqslant l$, this is equivalent to the existence of a number $s \leqslant l + 1$ such that $|U| \geqslant s$ and $|U \cap \{a_1, \ldots, a_l\}| < s$. Now observe that for any fixed $s \leqslant l + 1$, the condition $|U| \geqslant s$ can be expressed by a basic local sentence γ_s (with parameters $r = 0$ and s) and the condition $|U \cap \{a_1, \ldots, a_l\}| < s$ can be expressed by a 0-local formula $\psi_s(x_1, \ldots, x_l)$.

Example 9.15. This example extends the previous one, and illustrates the core machanics of the proof of Theorem 9.9. Consider graphs equipped with a unary predicate U, and let $\varphi(x_1, \ldots, x_l)$ express that there is some vertex y satisfying U whose distance to either of $x_1, \ldots, x_l$ is larger than 1:

$$\varphi(\bar{x}) \equiv \exists y.\, U(y) \wedge (\mathrm{dist}(y, x_1) > 1) \wedge \cdots \wedge (\mathrm{dist}(y, x_l) > 1).$$

Note that $\varphi(\bar{x})$ is not an r-local formula, for any fixed number r. We will show that $\varphi(\bar{x})$ is equivalent to a boolean combination of r-local formulas $\psi(\bar{x})$ and basic local sentences γ, for some r.

Fix a graph G with a distinguished set $U \subseteq V(G)$, and a tuple $a_1, \ldots, a_l$ of vertices. We want to determine whether $\varphi(a_1, \ldots, a_l)$ holds basing only on some local properties around $a_1, \ldots, a_l$, as well as some global properties of (G, U), independent of $a_1, \ldots, a_l$.

The first idea that could come to mind is to compare the number $n := |U|$ with the number $n' := N_1(\bar{a}) \cap U$, where $N_1(\bar{a})$ is the set of vertices at distance at most 1 from either of $a_1, \ldots, a_l$ in G. Then $\varphi(\bar{a})$ is equivalent to saying that $n < n'$ (at least in finite graphs). However, using a fixed first-order formula we may count only up to a fixed threshold k, and the numbers n and n' may be arbitrarily large (or even infinite, in infinite graphs).

Instead of counting all vertices in $U \cap N_1(\bar{a})$, let us now only look at vertices that are far apart. More precisely, a set of vertices $S \subseteq U$ is *scattered* (in G) if the mutual distance between any two elements of S is greater than 2 in G.

The key observation is that, although $N_1(\bar{a}) \cap U$ may be arbitrarily large, every scattered set contained in it has at most l elements. Indeed, if $S \subseteq N_1(\bar{a})$

is scattered then $S \subseteq \bigcup_{i=1}^{l} N_1(a_i)$ and moreover, no two elements of S belong to the same ball $N_1(a_i)$, since otherwise their distance would be at most 2.

Now, we compare the maximal size m of a scattered set contained in U with the maximal size m' of a scattered set contained in $U \cap N_1(\bar{a})$. We show that for all $a_1, \ldots, a_l \in V(G)$, the following conditions are equivalent:

1. There is some $b \in U$ such that $\mathrm{dist}(b, \bar{a}) > 1$ (that is, $G \models \varphi(\bar{a})$),
2. At least one of two conditions holds:
   (a) there is some $b \in U$ such that $1 < \mathrm{dist}(b, \bar{a}) \leqslant 3$, or
   (b) the maximal size m of a scattered subset of U is strictly larger than the maximal size m' of a scattered subset of $U \cap N_1(\bar{a})$.

We first prove (1)→(2). Suppose there is some $b \in U$ such that $\mathrm{dist}(b, \bar{a}) > 1$. Let S be a scattered subset of $U \cap N_1(\bar{a})$ of maximal size. If $\mathrm{dist}(b, \bar{a}) \leqslant 3$ then (2a) holds. Otherwise, $\mathrm{dist}(b, c) > 2$ for all $c \in S$, since $\mathrm{dist}(b, c) \leqslant 2$ would imply $\mathrm{dist}(b, a_i) \leqslant 3$ for some i. Therefore, $S \cup \{b\}$ is a scattered set of size $|S| + 1$, so (2b) holds.

We now prove (2)→(1). Clearly, (2a) implies (1), so assume (2b) holds. Let $M \subseteq V(G)$ be a scattered subset of U of maximal size. Then $S := M \cap N_1(\bar{a})$ is also scattered. By (2b), $S \neq M$, so there is some $b \in U - N_1(\bar{a})$, proving (1).

This proves the equivalence of (1) and (2). We now use (2) to rewrite φ into a boolean combination of r-local formulas and basic local sentences, for some r. Clearly, condition (2a) can be expressed using a 3-local formula. We now turn to condition (2b), and recall that every scattered subset of $N_1(\bar{a}) \cap U$ has size at most l. Therefore, condition (2b) is equivalent to a disjunction, over $0 \leqslant s \leqslant l + 1$, of conjunctions of two formulas:

- a sentence γ_s expressing that U contains a scattered set of size s,
- a formula $\psi_s(x_1, \ldots, x_l)$ expressing that $U \cap N_1(\{a_1, \ldots, a_l\})$ does not contain a scattered set of size s.

The sentence γ_s is in fact a basic local sentence with parameters $r = 1$ and s, whereas the formula $\psi_s(x_1, \ldots, x_l)$ is 1-local.

Summing up, $\varphi(x_1, \ldots, x_l)$ is equivalent to a boolean combination of 3-local formulas and basic local sentences with parameters $r = 1$ and $s = l + 1$.

We now prove Theorem 9.9 in general.

Proof of Theorem 9.9. The proof proceeds by induction on the structure of the formula. In the induction base, $\varphi(\bar{x})$ is an atomic formula, and is therefore 0-local. In the inductive step, the case of Boolean combinations is trivial. The nontrival case is that of existential quantification, that is, when considering a formula of the form $\exists y.\varphi(\bar{x}, y)$. By inductive assumption, we may assume that $\varphi(\bar{x}, y)$ is r-local (the basic local sentences may be pushed outside of the quantifier).

We now aim to translate the formula $\exists y.\varphi$ again into a local formula, possibly with a larger locality radius. Clearly, $\exists y.\varphi$ is equivalent to the disjunction of the following two formulas:

$$\exists y.\, (\mathrm{dist}(\bar{x}, y) \leqslant 2r + 1) \wedge \varphi(\bar{x}, y)$$
$$\exists y.\, (\mathrm{dist}(\bar{x}, y) > 2r + 1) \wedge \varphi(\bar{x}, y). \tag{9.1}$$

The first disjunct is a $(3r + 1)$-local formula, as desired. For the second disjunct, we use the following.

Lemma 9.16. *There is a finite set Δ of pairs of r-local formulas $(\alpha(\bar{x}), \beta(y))$ such that the following conditions are equivalent for any structure $\mathbb{A}$ and tuple $\bar{a}b \in \mathbb{A}^{\bar{x}y}$ with* $\mathrm{dist}(\bar{a}, b) > 2r + 1$*:*

- $\mathbb{A} \models \varphi(\bar{a}b)$
- $\mathbb{A} \models \alpha(\bar{a}) \wedge \beta(b)$ *for some* $(\alpha, \beta) \in \Delta$.

Proof. See Exercise 9.17. ∎

By Lemma 9.16, the disjunct (9.1) is equivalent to a finite disjunction of formulas

$$\exists y.\, (\mathrm{dist}(\bar{x}, y) > 2r + 1) \wedge \alpha(\bar{x}) \wedge \beta(y),$$

where $\alpha(\bar{x})$ and $\beta(y)$ are r-local, which in turn is equivalent to

$$\alpha(\bar{x}) \wedge \exists y. (\mathrm{dist}(\bar{x}, y) > 2r+1) \wedge \beta(y).$$

So it remains to show that the formula

$$\gamma(\bar{x}) \quad = \quad \exists y. (\mathrm{dist}(\bar{x}, y) > 2r+1) \wedge \beta(y) \tag{9.2}$$

is equivalent to a Boolean combination of local formulas and basic local sentences.

The formula (9.2) is very similar to the formula considered in Example 9.15. We may formally reduce the formula (9.2) to the formula considered in the example, by considering the relation $E(x, y)$ defined by the relation $\mathrm{dist}(x, y) \leqslant 2r+1$, and the unary predicate $U(x)$ defined by the formula $\beta(x)$. The formula $\gamma(\bar{x})$ obtained in Example 9.15 is a boolean combination of formulas of one of three forms:

- a formula $\alpha(\bar{x})$ expressing that there is some y satisfying $\beta(y)$ with $1 < \mathrm{dist}_E(\bar{x}, y) \leqslant 3$. Here dist_E, is the distance measured with respect to E, and therefore, in the Gaifman graph of the original structure we have $\mathrm{dist}(x, y) = (2r+1) \cdot \mathrm{dist}_E(x, y)$. Therefore, $\alpha(\bar{x})$ is r'-local with $r' = (6r+3) + r = 7r+3$, where the additional r comes from the fact that β is r-local.

- a sentence expressing that there is a set S of s elements satisfying $\beta(y)$ such that $\mathrm{dist}(x, y) > 2(2r+1)$ for all $x, y \in S$, where $s \leqslant |\bar{x}| + 1$,

- formulas expressing that $N_{2r+1}(\bar{x})$ does not contain a set S of s elements satisfying $\beta(y)$ such that $\mathrm{dist}(x, y) > 2(2r+1)$ for all $x, y \in S$, where $s \leqslant |\bar{x}| + 1$.

All this is a boolean combination of r'-local formulas and basic local sentences with parameters $r' = 7r+3$ and $s \leqslant |\bar{x}| + 1$.

This ends the inductive proof. A straightforward analysis of the proof leads to a bound on the end locality radius which is slightly worse than 7^q, where q is the quantifier rank of the original formula. However, a refinement of this analysis yields the bound 7^q. We omit the details. ∎

Exercise 9.17. Prove Lemma 9.16, using Corollary 8.2.

9.1 Model-checking on graphs of bounded degree

As an application of Gaifman's theorem we get an algorithm for model-checking first-order logic which runs in linear time on graphs of bounded degree.

Theorem 9.18 (Seese). *Fix $d \in \mathbb{N}$ and a first-order sentence φ. There is an algorithm which determines if a given graph G of maximum degree d satisfies φ in time $\mathcal{O}_{d,\varphi}(|G|)$.*

Above $|G|$ denotes the number of vertices of G. We assume that G is represented by the adjacency list representation, that is, by listing all the edges of G. Note that the number of edges is at most $d \cdot |G| = \mathcal{O}_d(|G|)$.

The result above also holds (with the same proof) for relational structures whose Gaifman graph has maximum degree bounded by a constant d.

Proof. By Theorem 9.9, we may assume that φ is a Boolean combination of basic local sentences. This reduces further to the case when φ is a single basic local sentence, expressing that there is an r-scattered set of size s whose elements satisfy an r-local formula $\psi(x)$.

In a graph G with maximum degree d, each neighborhood of radius r has size at most $d^{r+1} + 1 = \mathcal{O}_{r,d}(1)$. Hence, testing whether a given vertex $v \in G$ satisfies the r-local formula $\psi(x)$ can be done in time $\mathcal{O}_{d,r,\psi}(1)$ independent of G (more precisely, in time $(d^{(r+1)+1})^{|\psi|}$, by Proposition 3.2). Therefore, in time $\mathcal{O}_{d,r,\psi}(|G|)$, we may mark the set R of all the vertices in G which satisfy $\psi(x)$.

It remains to test whether R contains an r-scattered set of size s, in time $\mathcal{O}_{d,r,s}(|G|)$.

Construct the r-scattered set $S \subseteq R$ greedily. Start with $S = \emptyset$. Iterate through all vertices $v \in R$, and add v to S if $\mathrm{dist}(v, S) > r$. Terminate once $|S| = s$ or all the vertices $v \in R$ have been examined.

Testing whether $\mathrm{dist}(v, S) > r$ can be done in time $d^{r+1} = \mathcal{O}_{d,r}(1)$, by exploring the entire neighborhood of v of radius r. Hence, the above procedure takes time $\mathcal{O}_{d,r}(|G|)$, and terminates with one of two outcomes:

1. an r-scattered set S of size s,
2. a set S with $|S| < s$ such that $R \subseteq N_r(S)$.

In the first case, we are done. In the second case, R is contained in a set of size at most $s \cdot d^{r+1} = \mathcal{O}_{d,r,s}(1)$, so the existence of an s-scattered set of size s can be done in R using brute-force in time $\mathcal{O}_{d,r,s}(1)$.

Altogether, this yields an algorithm which determines the existence of an r-scattered subset of R of size s in time $\mathcal{O}_{d,r,s}(|G|)$. ∎

9.2 *Bounded local treewidth*

Say that a graph G has radius at most r if there is a vertex v such that every $w \in G$ is within distance r from v.

The notion of local treewidth is motivated by the following observation.

Theorem 9.19. *A connected planar graph of radius r has treewidth at most $3r$.*

Proof. Consider an embedding of G into the plane. Since the treewidth cannot decrease while adding an edge, we can assume that G is a triangulation, that is, the boundary of every face consists of three edges.

As G has radius r, there is a vertex v and a spanning tree T rooted at v in which every root-to-leaf path has at most $r+1$ vertices. The key observation (cf. Fig. 9.1) is that the faces form a tree in which two faces are adjacent if they share an edge of $G - T$. Define the bag of a face f as the set of vertices along the three paths in T from u to either vertex incident to f in T. This yields a tree decomposition of G of width at most $3r$. ∎

Let $\mathcal{C}$ be a class of graphs which is closed under taking subgraphs. Say that a class of graphs $\mathcal{C}$ has *bounded local treewidth* if there is a function $f\colon \mathbb{N} \to \mathbb{N}$ such that for every $G \in \mathcal{C}$,

$$\text{treewidth}(G) \leqslant f(\text{radius}(G)).$$

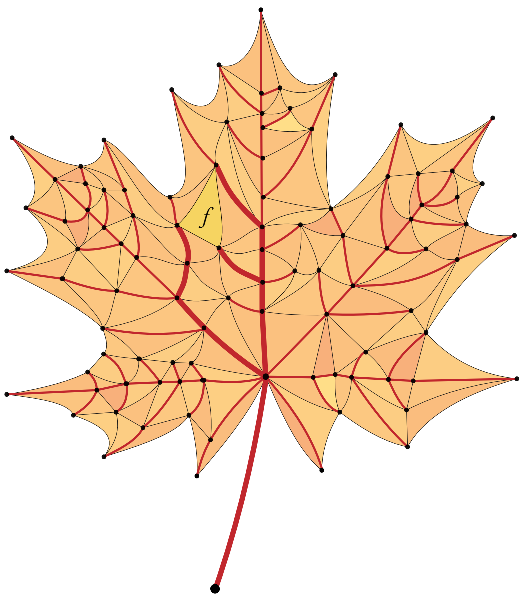


Figure 9.1: A planar graph G with a fixed embedding into the plane and a rooted spanning tree T (red edges). The faces (including the outer face) form a tree, in which two faces are adjacent if they share an edge of $G - T$. Define the bag of a face f as all the vertices along the paths in T from the root to either of the vertices adjacent to f. This gives a tree decomposition of G. If T has depth r and G is a triangulation then each bag has at most $3r + 1$ vertices, so treewidth$(G) \leqslant 3r$.

Example 9.20. The class of trees has bounded local treewidth with function $f(r) = 1$ for $r \in \mathbb{N}$. The class of graphs of treewidth c has bounded local treewidth with function $f(r) = c$ for $r \in \mathbb{N}$. The class of graphs with maximum degree c has bounded local treewidth with function $f(r) = c^{r+1}$. The class of planar graphs has bounded local treewidth with function $f(r) = 3r$. More generally, for every closed surface S, the class of graphs which embed into S has bounded local treewidth.

Theorem 9.21. *Fix a first-order sentence φ and a class $\mathcal{C}$ with bounded local treewidth. There is an algorithm which given a graph $G \in \mathcal{C}$ decides if G satisfies φ in time $\mathcal{O}_\varphi(|G|^2)$.*

Proof. By Gaifman's locality theorem, Theorem 9.9, we can assume that φ expresses the existence of an r-scattered set of size s of elements satisfying an r-local formula $\alpha(x)$.

For each vertex v of G, color v red if

$$G[N_r(v)] \models \alpha(v).$$

As $G[N_r(v)]$ has radius r, we have that $G[N_r(v)]$ has treewidth at most $f(r)$, where f is the function witnessing that $\mathcal{C}$ has bounded local treewidth. Using Theorem 8.20, the color of a given vertex v can be determined in time $\mathcal{O}_r(|G|)$. The total runtime required to color all vertices is then $\mathcal{O}_r(|G|^2)$.

It remains to determine whether G has an r-scattered set consisting of s red vertices. To this end, we greedily construct a sequence of red vertices $v_1, v_2, \ldots$, in each step picking any red vertex v_n whose distance to each of $v_1, \ldots, v_{n-1}$ is larger than $2r$. If we succeed in constructing the sequence for s steps, then we are done. Otherwise, we have a set $v_1, \ldots, v_k$ with $k < s$ such that every red vertex v is in distance at most $2r$ from one of $v_1, \ldots, v_k$.

Group the vertices $v_1, \ldots, v_k$ into clusters – the connected components of the relation of being in distance at most $6r$. For each cluster C_i, let $N_i \subseteq V(G)$ denote the set of vertices in distance at most $2r$ from a vertex in C_i.

Then:

- $N_1 \cup \cdots \cup N_l$ contains all the red vertices in G,
- for $1 \leqslant i < j \leqslant l$ and for every red $v \in N_i$ and red $w \in N_j$, the distance between v and w in G is larger than $2r$;
- $G[N_i]$ is a subgraph of G of radius at most $6rs$, for each $1 \leqslant i \leqslant l$.

By the first property, an r-scattered set S of red vertices in G is the union of r-scattered sets $S \cap N_1, \ldots, S \cap N_l$. Moreover, by the second property, if S_i is an r-scattered set contained in $S \cap N_i$, for $i = 1, \dots, l$, then $S_1 \cup \cdots \cup S_l$ is an r-scattered set.

Hence, whether or not G contains an r-scattered set of s red vertices depends only on the information, for each $i = 1, \dots, l$ and $s' = 1, \dots, s$, whether $G[N_i]$ contains an r-scattered set of s' red vertices. As $G[N_i]$ has radius at most $6rs$, this information can be determined in time $\mathcal{O}_{r,s}(|G|)$ using Theorem 8.20 once again. ∎

Together with Theorem 9.19, we get:

Corollary 9.22. *Model-checking first-order logic can be solved in quadratic time in data complexity on all planar graphs.*

The exponent 2 in the running time in Theorem 9.21 can be replaced by any fixed real larger than 1, but this requires a more involved analysis. For graph classes which are closed under taking minors, such as the class of planar graphs, this can be further improved to linear time.

Replacing treewidth with clique-width, we get an analogous result, but where the exponent 2 is replaced by 4 (and can be further decreased to any real larger than 3), due to the use of Theorem 8.22. Say that a class of graphs $\mathcal{C}$ has *bounded local clique-width* if there is a function $f\colon \mathbb{N} \to \mathbb{N}$ such that for every $G \in \mathcal{C}$,

$$\text{clique-width}(G) \leqslant f(\text{radius}(G)).$$

Theorem 9.23. *Fix a first-order sentence φ and a class $\mathcal{C}$ with bounded local clique-width. There is an algorithm which given a graph $G \in \mathcal{C}$ decides if G satisfies φ in time $\mathcal{O}_\varphi(|G|^4)$.*

10
Descriptive complexity

The aim of *descriptive complexity* is to describe computational complexity classes by means of logics, and not by machines and their resources. The archetypal example of a result in this direction is Fagin's theorem, which states that every sentence φ of existential second-order logic (ESO) defines a problem which is in the complexity class NP, and conversely, every problem in NP concerning relational structures can be expressed by a sentence of ESO. Hence, NP corresponds precisely to the logic ESO, and ESO provides a machine-independent characterisation of NP. Descriptive complexity tries to find characterisations of other complexity classes by means of other logics, most importantly, for P. One of the key difficulties lies in the fact that while Turing machines (and other computation models) have access to a linear order on an input graph (as the graph is presented in some order), the logics do not.

10.1 Fagin's theorem

Fix a signature Σ. A *property* P of finite structures over Σ is a class of structures over this signature that is invariant under isomorphism: if two structures are isomorphic, then either both belong to P, or both do not belong to P. To represent a structure $\mathbb{A}$ as a word, we need to fix an ordering $\leqslant$ of its domain. When such an ordering is fixed, then the domain of $\mathbb{A}$ can be identified with $\{0, 1, \ldots, |\mathbb{A}| - 1\}$ in increasing order with respect to $\leqslant$. Let $\mathrm{enc}(\mathbb{A}, \leqslant)$ denote the adjacency matrix encoding of the corresponding structure with domain $\{0,1,\dots,|\mathbb{A}|-1\}$, as described in Section 3.1. Note that a structure $\mathbb{A}$ has many encodings, depending on the chosen order $\leqslant$. We do not fix any particular order, and consider all possible orderings. Then a property P induces the following language of words:

$$L_P = \{\mathrm{enc}(\mathbb{A},\leqslant) \mid \mathbb{A} \in P,\ \leqslant \text{ order on } \mathbb{A}\}\,.$$

The computational complexity of the property P is the computational complexity of the language L_P.

Example 10.1. Consider the graph signature $\{E\}$. The property "G is a two-colorable undirected graph" is in P, while the property "G is a three-colorable undirected graph" is complete for NP.

Recall that a formula of existential second-order logic (ESO) is a formula of the form

$$\exists R_1 \exists R_2 \dots \exists R_n . \varphi,$$

where $R_1,\dots,R_n$ are relation symbols and φ is a first-order formula which may use the symbols $R_1,\dots,R_n$.

Example 10.2. The following formula holds in a graph G if it is 3-colorable. It is monadic – the symbols R, G, B are unary.

$$\begin{aligned}&\exists R \exists G \exists B . \forall x . part(R,G,B) \wedge \forall x \forall y .\\ &((x \in R \wedge y \in R) \vee (x \in G \wedge y \in G) \vee (x \in B \wedge y \in B)) \rightarrow \neg E(x,y),\end{aligned}$$

where $part(R,G,B)$ expresses that the predicates R,G,B form a partition of the universe.

The main result of this chapter is:

Theorem 10.3 (Fagin). *Fix a signature Σ. A property of finite structures is ESO-definable if and only if it is in* NP.

One direction is easy: any ESO-definable property can be decided by a non-deterministic Turing machine, in polynomial time: first guess the relations –

each of them has polynomial size – and then evaluate the underlying first-order formula in polynomial time on the expanded structure.

Therefore, it remains to prove that any property which can be decided in NP, can be defined by an ESO formula.

Example 10.4. We show that the property "G is connected" is definable in ESO. To this end, observe that G is connected if and only if it has a spanning tree. So our ESO formula can start by saying "there is a relation T which represents a spanning tree". A spanning tree is a directed graph T whose edges are contained in the edges in G, such that apart from one vertex (the root) every vertex of G has exactly one incoming edge in T, and whose underlying graph is acyclic. The first two parts can be expressed by a first-order formula. To express acyclicity of T, note that it equivalent to the existence of a total order $\leqslant$ on its vertices such that every directed edge leads from a smaller vertex to a larger vertex. So finally, our formula is:

$$\exists \leqslant^{(2)} . \exists T^{(2)} . \bigwedge \begin{cases} \forall u, v. T(u,v) \to ((u < v) \wedge E(u,v)) \\ \exists ! x. \forall y. \neg T(y,x) \\ \forall x,y,z. (T(y,x) \wedge T(z,x)) \to (y = z) \end{cases}$$

Exercise 10.5. Non-connectivity can be expressed in existential monadic second-order logic. Prove that connectivity cannot.

Before proving Fagin's theorem, we introduce some useful notation.

Relations and formulas with values in finite sets. We make the following convention. Let Q be a finite set. If A is a set and k a natural number, then a Q-valued relation on A is a relation $R \subseteq A^k \times Q$, and can be represented by a tuple $(R_q : q \in Q)$ of relations of arity k on A:

$$R(\bar{x}, q) \equiv R_q(\bar{x}).$$

(There is a more efficient encoding, using $O(\log(|Q|))$ relations, but we will not need it).

We will therefore allow relational symbols for such Q-valued relations of arity k (where Q and k are fixed and finite).

In the same way, we may consider Q-valued formulas, which are just Q-tuples of formulas. We will use the following syntax for a Q-valued formula with values in $Q = \{p, q, r, \ldots\}$:

$$\begin{bmatrix} p: & \varphi_p \\ q: & \varphi_q \\ r: & \varphi_r \\ \ldots & \ldots \end{bmatrix}.$$

The free variables of a Q-valued formula are all the free variables of its constituents. The semantics of a Q-valued formula with k free variables is a Q-valued relation of arity k. Therefore, if φ is a Q-valued formula with free variables $\bar{x}$, then $\varphi(\bar{x}, q)$ is equivalent to $\varphi_q(\bar{x})$.

Observe that the properties that a Q-valued relation R is a function, or a partial function $R : \mathbb{A}^k \to Q$, can be expressed by first-order formulas. If f is a Q valued partial function, then we write $f(\bar{x}) = q$ for $f(\bar{x}, q)$ (which in turn is encoded as $f_q(\bar{x})$).

Proof of Theorem 10.3. We prove the "only if" in Fagin's theorem. Consider any property P of finite structures which is decidable by a nondeterministic Turing machine M in polynomial time, so that that the language L_P is decidable in NP.

Let B be the work alphabet and Q be the state space of the machine M. Suppose that k is a natural number such that an input structure of cardinality $n > 1$ is accepted by some run of M of length at most n^k. Moreover, we suppose that k is greater than the arity of the relations in the signature.

We construct an ESO formula which will accept $\mathbb{A}$ if and only if for some linear order $\leqslant$ on $\mathbb{A}$, the machine M accepts the encoding of $(\mathbb{A}, \leqslant)$ by its adjacency matrix. The formula will guess:

- A binary relation $\leqslant$, intended to represent a linear order,
- A relation f of arity $2k$ with values in $B \cup B \times Q$, intended to represent an accepting run on $\mathrm{enc}(\mathbb{A}, \leqslant)$.

Firstly, we use an FO formula to assert that $\leqslant$ is a linear ordering of the universe. For two tuples $\bar{a}, \bar{b} \in \mathbb{A}^s$, when writing $\bar{a} \leqslant_{\mathrm{lex}} \bar{b}$, we mean the lexicographic order on $\mathbb{A}^k$ with respect to $\leqslant$. Observe that for a fixed number s, this order is definable by a first-order formula using $\leqslant$.

Secondly, we use an FO formula to assert that f is a partial function (with values in $B \cup B \times Q$).

Thirdly, we verify that f encodes a run of the Turing machine in the following way. Let $\bar{x}$ and $\bar{y}$ be, respectively, the ith and jth lexicographically smallest tuples in $\mathbb{A}^k$. Then $f(\bar{x}, \bar{y}) = b$ is interpreted as:

the tape symbol in the ith configuration, at its jth position is b,

and $f(\bar{x}, \bar{y}) = (b, q)$ additionally asserts that

the head of the machine is located at this position in state q.

Testing that f indeed encodes a run of the machine M can be easily expressed in first-order logic, since the successor/predecessor of a tuple $\bar{x} \in \mathbb{A}^k$ (with respect to the lexicographic order) can be defined in first-order logic.

The last thing that remains is to check that the initial configuration of the run encoded by f corresponds to the encoding of $(\mathbb{A}, \leqslant)$. This can be done by a first-order formula, by the following lemma.

Lemma 10.6. *Fix $k \in \mathbb{N}$. There is a first-order formula $\varphi(x_1, \ldots, x_k)$ such that for every linearly ordered structure $(\mathbb{A}, \leqslant)$ and $1 \leqslant i \leqslant |\mathbb{A}|^k$, the ith smallest tuple $\bar{a}$ with respect to $\leqslant_{lex}$, $\bar{a} \in \mathbb{A}^k$, satisfies $(\mathbb{A}, \leqslant) \models \varphi(\bar{a})$ if and only the ith bit of enc$(\mathbb{A}, \leqslant)$ is* 1.

Proof. Let $n = |\mathbb{A}|$. For tuples $\bar{a}, \bar{b} \in \mathbb{A}^k$ and $i \in \mathbb{N}$ write $\bar{a} = \bar{b} + i$ to denote that $\bar{b}$ is the i-fold successor of $\bar{a}$ with respect to the lexicographic order on $\mathbb{A}^k$. Note that for each fixed $i \leqslant k$, there is a formula $\varphi_i(\bar{x}, \bar{y})$ that holds of a pair $(\bar{a}, \bar{b}) \in \mathbb{A}^{2k}$ if and only if $\bar{b} = \bar{a} + n^i$. This is verified by performing addition in base n of two numbers of length k.

Let $\Sigma = \{R_1, \ldots, R_\ell\}$ where R_i has arity r_i for $i = 1, \ldots, \ell$. Let $\bar{z} \in \mathbb{A}^k$ be the lexicographically least tuple. We have:

- $\neg\varphi(\bar{z}+n)$ and $\varphi(\bar{a})$ for all $\bar{a} \leqslant_{\text{lex}} \bar{z}+n$;
- for $i = 1, \ldots, |\Sigma|$ and for each $\bar{a} \in \{0\}^{k-r_i} \times \mathbb{A}^{r_i}$,

$$\varphi(\bar{a}+n+1+n^{r_1}+n^{r_2}+\cdots+n^{r_{i-1}}) \qquad \Longleftrightarrow \qquad \mathbb{A} \models R_i(\bar{a}).$$

- $\neg\varphi(\bar{a})$ for all tuples $\bar{a} \in \mathbb{A}^k$ larger than $\bar{z}+n+1+n^{r_1}+n^{r_2}+\cdots+n^{r_{i_\ell}}$.

Each of the above conditions can be verified by a first-order formula. ■

To conclude, the ESO expressing that M accepts an encoding of the structure $\mathbb{A}$ is of the form:

$$\exists\leqslant\exists f. \begin{cases} \leqslant & \text{is a linear order} \\ f & \text{is a function of arity } 2k \text{ with values in } B \cup B \times Q \\ f & \text{encodes a run of } M \\ & \text{whose first configuration is the encoding of } (\mathbb{A}, \leqslant) \\ & \text{an accepting state is reached} \end{cases}$$

This finishes the proof of Fagin's Theorem. ■

Exercise 10.7. Deduce from Fagin's Theorem the Cook-Levin Theorem, that SAT is NP-complete.

10.2 Fixpoint logics

In view of Fagin's Theorem, the following question arises.

> **The big question:** Does there exist a logic $\mathcal{L}$ such that a property of finite structures is definable in the logic $\mathcal{L}$ if and only if it is in P?

We will see that fixpoint logic offers a partial answer to this question. Before giving the definitions, let's see an example.

Example 10.8. Consider the signature of directed graphs. The following formula, with free variables x, y, says that there is a directed path from x to y:

$$\begin{aligned} \varphi(x,y) = \text{let } R(z) = \text{ifp } & [(z = x) \vee \exists v. R(v) \wedge E(v,z)] \qquad (\diamond) \\ \text{in } & R(y). \end{aligned}$$

The semantics is as follows. Let x, y be vertices of a graph. Start with R being the empty set of vertices, treated as a unary relation. Repeat indefinitely the following inflationary step:

Add to R those vertices z which satisfy the subformula of $(\diamond)$ delimited by the brackets [ ].

The formula $\varphi(x,y)$ holds for the pair of vertices (x,y) if y belongs to R in some step of the loop. Observe that the property described above is not first-order definable.

Fixpoints

Fix a signature Σ and let R be an additional symbol for a k-ary relation with values in a finite set Q.

Let φ be a formula with values in Q, with k distinguished free variables $\bar{x}$, over the signature $\Sigma \cup \{R\}$. Then φ defines a mapping of k-ary, Q-valued relations over $\mathbb{A}$, i.e. a self-mapping of the set $P(\mathbb{A}^k \times Q)$:

$$F_\varphi(\mathcal{R}) = \{\bar{a} \in \mathbb{A}^k : \mathbb{A}, \mathcal{R}, \bar{a} \models \varphi\}.$$

A *fixpoint* of a mapping $F : P(\mathbb{A}^k \times Q) \to P(\mathbb{A}^k \times Q)$ is a relation $\mathcal{R}$ such that $F(\mathcal{R}) = \mathcal{R}$. In general, a mapping may have none or several fixpoints.

Suppose that F is monotone, i.e., $\mathcal{R} \subseteq \mathcal{S} \to F(\mathcal{R}) \subseteq F(\mathcal{S})$ for all relations $\mathcal{R}, \mathcal{S}$. Then, by the Knaster-Tarski theorem, F has a least fixpoint, denoted lfp F and a greatest fixpoint, denoted gfp F. Indeed, for the least fixpoint we consider the sequence $F^\alpha(\emptyset)$ defined by transfinite induction for all ordinals α (as usually: $F^\alpha = F \circ F^{\alpha-1}$ if $\alpha \geqslant 1$ is a successor ordinal and $F^\alpha(\mathcal{R}) = \bigcup_{\beta<\alpha} F^\beta(\mathcal{R})$ if α is a limit point). If F is monotone, then the sequence is increasing and reaches a fixpoint for some ordinal α.

Suppose now that F is *increasing*, i.e., $\mathcal{R} \subseteq F(\mathcal{R})$ for all $\mathcal{R}$. Then the sequence $F^{\alpha}(\emptyset)$ is also increasing and reaches a fixpoint (possibly after an infinite ordinal number of steps). Note that this fixpoint need not be the least fixpoint of F. Even if G is a mapping which is not necessarily increasing, then the mapping F given by $\mathcal{R} \mapsto G(\mathcal{R}) \cup \mathcal{R}$ is clearly increasing. We call the fixpoint of the mapping F the *inflationary* fixpoint of the mapping G. Warning: it does not need to be a fixpoint of G.

Observe that if $\mathbb{A}, Q$ and k are finite, then an increasing sequence of subsets of $\mathbb{A}^k \times Q$ reaches a fixpoint after at most $|\mathbb{A}|^k \cdot |Q|$ steps, i.e., polynomially many in terms of $|\mathbb{A}|$ (for fixed Q and k).

Inflationary Fixpoint Logic

Inflationary Fixpoint Logic (IFP) is an extension of first-order logic by a fixpoint operation whose syntax is the following:

$$\begin{aligned} &\text{let } R(\bar{x}) = \text{ifp } \varphi \\ &\text{ in } \psi, \end{aligned} \tag{γ}$$

where:

- R is a Q-valued k-ary relation, where Q is some finite set and k is some finite number;
- φ is a Q-valued formula, over the signature of $\mathbb{A}$ extended by the symbol R. Recall that formally, φ is a tuple $(\varphi_q)_{q \in Q}$ of formulas;
- $\bar{x}$ is a finite set of variables.

The free variables of γ are those free variables of φ which do not occur in $\bar{x}$, together with the free variables of the formula ψ.

The semantics is defined so that the formula γ holds in a structure $\mathbb{A}$, with given valuations for its free variables, iff the formula ψ holds in the structure $\mathbb{A}$ extended by the relation ifp F_{φ} as the interpretation for the symbol R. Note that in the fixpoint, the mapping F_φ may depend on several parameters – those free variables of φ which do not occur in the tuple $\bar{x}$. Note that the values of those parameters are fixed when computing the fixpoint.

A formula of IFP may use the above fixpoint operation – possibly in a nested way – apart from the usual constructs of first-order logic.

The example given earlier demonstrates that the logic IFP is strictly more expressive than first-order logic, since reachability cannot be expressed in FO. It turns out that over linearly-ordered structures, IFP is as expressive as polynomial time Turing machines.

Exercise 10.9. Show that two-colorability can be decided by a formula of IFP, by simulating the greedy algorithm.

The following proposition follows easily from the fact that a fixpoint must be reached in a number of steps bounded by a polynomial in the size of the considered structure.

Proposition 10.10. *Let φ be a formula of IFP. There is an algorithm which given a finite structure $\mathbb{A}$ decides whether it satisfies φ in time polynomial in $|\mathbb{A}|$.*

So every property definable in IFP is in P. The converse fails for trivial reasons: there is no formula of IFP which determines the parity of a given set.

Exercise 10.11. Show this.

However, for ordered structures, IFP corresponds precisely to P. This is stated below.

Theorem 10.12 (Immerman-Vardi)**.** *A property of linearly-ordered structures is definable in the logic IFP if and only if this property is in P.*

Proof. As usual, one direction is easy: from logic to machines. Indeed, let φ be an IFP formula of linearly ordered structures. Then, for a given description of a structure $\mathbb{A}$, a Turing machine can evaluate φ over $\mathbb{A}$ in polynomial time. Since the mapping $\mathcal{R} \mapsto \mathcal{R}'$ is monotone (see definition of ifp), it follows that it stabilizes after at most n^k steps, where k is the arity of $\mathcal{R}$ and n is the size of $\mathbb{A}$. Therefore, the computation of the least fixpoint can be executed in polynomial time (the parameter k depends on the formula φ, but not on the input structure $\mathbb{A}$).

We show the other implication, by following the idea in the proof of Fagin's theorem. Suppose that M is a Turing machine which recognises a property of linearly-ordered structures in polynomial time. We assume that M expects the adjacency matrix encoding of a structure $(\mathbb{A}, <)$ on input. Let B be the work alphabet and Q be the state space of the machine M. Assume without loss of generality that the machine only accepts when in state q_{fin} and the current letter is $b \in B$. Let k be such a number that M works in time at most n^k for structures of size $n \geqslant 2$. We will use f as a symbol of a k-ary relation (in fact, a function) with values in $B \cup B \times Q$. The formula which holds in $(\mathbb{A}, <)$ iff M accepts the encoding of $(\mathbb{A}, <)$ will have the following form:

$$\begin{aligned}&\text{let } f(\bar{x}, \bar{y}) = \text{ifp}[\varphi_1(\bar{x}, \bar{y}) \vee \varphi_2(\bar{x}, \bar{y})] \\ &\ \text{in } \exists \bar{x}\bar{y}. f(\bar{x}, \bar{y}) = (b, q_{fin}).\end{aligned}$$

We need to explain what the formulas φ_1 and φ_2. As in the proof of Fagin's theorem, f is intended to be a function encoding a run of the Turing machine, where $f(\bar{x}, \bar{y}) = b$ is interpreted as:

- the tape symbol in the $[x]$th configuration, at its $[y]$th position is b,
- and $f(\bar{x}, \bar{y}) = (b, q)$ additionally asserts that the head of the machine is located at this position in state q.

Hence, $f(\bar{x}, \bar{y}) = (b, q_{fin})$ means that an accepting configuration is reached.

The formulas $\varphi_1(\bar{x}, \bar{y})$ and $\varphi_2(\bar{x}, \bar{y})$ are $B \cup B \times Q$-valued formulas. The formula φ_1 encodes the structure $(\mathbb{A}, <)$ – the existence of such a formula was argued in the proof of Fagin's theorem. The formula $\varphi_2(\bar{x}, \bar{y})$ yields for given $\bar{x}, \bar{y}$ the value b or (b, q) depending on the values

$$f(\bar{x}_{-1}, \bar{y}_{-1}), \quad f(\bar{x}_{-1}, \bar{y}), \quad f(\bar{x}_{-1}, \bar{y}_{+1}),$$

where $\bar{z}_{\pm 1}$ denotes a tuple which encodes the number $[\bar{z}_{\pm 1}]$ which is the successor/predecessor of the number $[\bar{z}]$ encoded by $\bar{z}$ (see the proof of Fagin's theorem). Recall that there are first-order formulas $\varphi_{\pm 1}(\bar{z}, \bar{u})$ which hold if and only if $\bar{u} = \bar{z}_{\pm 1}$. This relies heavily on the given linear order.

The formula φ_2 has the form:

$$\begin{bmatrix} b : & \varphi_b \\ \cdots & \cdots \\ (b,q) : & \varphi_{b,q} \\ \cdots & \cdots \end{bmatrix},$$

where φ_b consists of a prefix

$$\exists \bar{x}_{-1}, \bar{y}_{-1}, \bar{y_{+1}}.\varphi_{-1}(\bar{x}, \bar{x}_{-1}) \wedge \varphi_{-1}(\bar{y}, \bar{y}_{-1}) \wedge \varphi_{+1}(\bar{y}, \bar{y}_{+1})$$

followed by a disjunction of formulas of the form

$$f(\bar{x}_{-1}, \bar{y}_{-1}) = c \wedge f(\bar{x}_{-1}, \bar{y}) = d \wedge \; f(\bar{x}_{+1}, \bar{y}_{-1}) = e,$$

where $c, d, e \in B \cup B \times Q$.

■

Least Fixpoint Logic

Another fixpoint logic is the *least fixpoint logic* (LFP) . Its syntax and semantics are similar to IFP, but ifp is replaced by lfp. However, there is one additional syntactic requirement. In order to assure that the mapping F_φ is monotone (and hence has a least fixpoint), it is required by the syntax that the symbol R occurs positively in φ, i.e., under the scope of an even number of negations.

Example 10.13. The formula from the previous example, with ifp replaced by lfp, becomes a valid formula of LFP logic.

Observe that in the formula constructed in the proof of the theorem, the symbol f appears positively. Therefore, the above proof also works for the LFP logic, giving the original formulation of the Immerman-Vardi theorem.

Theorem 10.14 (Immerman-Vardi, original form). *A property of linearly ordered structures is definable in the logic LFP if and only if this property is in* P.

The two variants of the theorem together give the following.

Corollary 10.15. *Over linearly ordered finite structures, the logics LFP and IFP are equally expressive.*

Note that converting a formula of LFP into an equivalent formula of IFP is immediate. Indeed, it is enough to substitute lfp by ifp. However, the conversion in other direction is not at all obvious, since in lfp the predicate needs to occur positively.

It is a theorem (of Gurevich and Shelah) that over all finite structures, IFP and LFP are equally expressive. Finally, Kreutzer proved that this also holds over all (finite and infinite) structures.

Relation to the bounded variable fragment

Fixpoint logics are closely related to the bounded variable fragment of FO. Intuitively, in a finite structure $\mathbb{A}$ a formula of LFP or IFP can be unravelled to a first-order formula (by just expanding the definition of the fixpoint operator) and this will yield a formula using a bounded number of variables, independent on n. This is expressed more precisely in the following.

Lemma 10.16. *Fix a sentence φ of LFP or IFP. There is a number k such that for every number n there is a sentence $\psi \in \mathrm{FO}^k$ such that φ and ψ are equivalent on all structures of size at most n.*

We skip the proof, which is rather syntactic.

Corollary 10.17. *If $\mathbb{A}$ and $\mathbb{B}$ are finite and $\mathbb{A} \models \varphi$ and $\mathbb{B} \models \neg\varphi$ then spoiler wins the bijective k-pebble game on $\mathbb{A}$ and $\mathbb{B}$.*

Proof. Pick n larger than $|\mathbb{A}|$ and $|\mathbb{B}|$ and apply Lemma 10.16 yielding a formula $\psi \in \mathrm{FO}^k$ such that $\mathbb{A} \models \psi$ and $\mathbb{B} \models \neg\psi$. By Theorem 6.5, spoiler wins the k-pebble game on $\mathbb{A}$ and $\mathbb{B}$. ∎

10.3 Fixpoint logic with counting

Recall that IFP cannot even express whether a given set has even size, or whether a given structure is the disjoint union of two cliques of equal size. So IFP fails to capture P already because of its inability to count.

The logic IFP+C is an extension of the logic IFP which allows to count, up to a polynomial threshold. For a finite structure $\mathbb{S}$ with n elements, let $[\mathbb{S}]$ denote the structure whose universe are the elements $0, 1, \ldots, n-1$, and which is equipped with two binary predicates: the linear order $\leqslant$ and the successor relation *succ*, whose symbols are distinct from the symbols used in $\mathbb{S}$. By $\mathbb{S} + [\mathbb{S}]$ we denote the two-sorted structure, extending $\mathbb{S}$ – whose elements are called nodes – by the counting sort $[\mathbb{S}]$, whose elements are called numbers. If φ is a formula and $\bar{x} = (x_1, \ldots, x_k)$ are some of its free variables ranging over nodes and $i \in \mathbb{N}$ then $\#_i\bar{x}.\varphi$ is the element of the counting sort which is the i-th digit of the number of tuples $\bar{x}$ of $\mathbb{S}$ which satisfy the formula φ, in the base-n encoding, where n is the cardinality of $\mathbb{S}$.

The syntax of IFP+C is defined by extending the syntax of IFP by the counting construct $\#_i\bar{x}.\varphi$, and also allowing to use the relations $\leqslant$ and *succ*. To interpret such a formula over a structure $\mathbb{S}$ means to interpret it over $\mathbb{S} + [\mathbb{S}]$, where the interpretation of the counting construct, the linear order and successor relation are as described above.

Example 10.18. To determine whether a pure set X has even cardinality, we use a formula ψ of IFP which holds in a linearly ordered structure if and only if it has even cardinality. The existence of such a formula follows from the Immerman-Vardi theorem, since parity is in P. It is also easy to explicitly write such a formula, by incrementally extending a unary predicate adding every second element. With the help of the formula ψ, we may define a formula of IFP+C which determines whether a pure set has even cardinality – it suffices to relativize ψ to elements ranging over the counting sort. Note that the elements of the counting sort are the only ones that satisfy the relation $x \leqslant x$. We haven't even used the counting construct – only the counting sort was used for determining the parity. Indeed, in case $\mathbb{S}$ is a pure set, the counting sort $[\mathbb{S}]$ is simply a richer version of $\mathbb{S}$, endowed additionally with the linear order, so in this case it makes sense to work directly with $[\mathbb{S}]$.

In a similar fashion, one can define any P-computable query of pure sets using IFP+C.

Example 10.19. We describe an IFP+C formula which computes the radius of a given graph G, i.e. the minimal value r such that there exists a node $v \in G$ whose r-neighborhood contains all of G. To this end, using IFP we define a ternary relation $D(u, v, r)$, where u and v range over nodes and r ranges through numbers, and such that the intended interpretation of $D(u, v, r)$ is $d(u, v) \leqslant r$. This can be easily done with IFP: we start with D containing all tuples $(u, u, 0)$, where $u \in G$, and in the inflationary step, we add to D tuples (u, v', r') such that there already is a tuple (u, v, r) in D such that v and v' are connected by an edge and r' is the successor of r. Using the relation $D(u, v, r)$, we define a formula $\varphi(r)$ as $\exists v.\forall u.D(u, v, r)$. Finally, we define a formula $\psi(r)$ as $\varphi(r) \wedge \neg\exists r' < r.\varphi(r')$. Then $\psi(r)$ holds if and only if G has radius r.

The last example still does not even use the counting construct $\#x.\varphi$; it only used the counting sort and its linear order.

Exercise 10.20. Define an IFP+C formula which holds in a graph if and only if has an even number of connected components. Hint: define a formula $N(a, b)$, where a, b range over numbers, which holds if and only if there are exactly a vertices whose connected component has cardinality b.

To prove inexpressibility results, it is useful to have a suitable notion of games, and this is provided by the link between IFP+C and the logic C. Just as IFP is related to the bounded variable fragment of first-order logic, IFP+C is related to the bounded variable fragment with counting.

Lemma 10.21. *Fix a sentence φ of IFP+C. There is a number k such that for every number n there is a sentence $\psi \in \mathrm{C}^k$ such that φ and ψ are equivalent on all structures of size at most n.*

Corollary 10.22. *If $\mathbb{A}$ and $\mathbb{B}$ are finite and $\mathbb{A} \models \varphi$ and $\mathbb{B} \models \neg\varphi$ then spoiler wins the bijective k-pebble game on $\mathbb{A}$ and $\mathbb{B}$.*

Proof. Pick n larger than $|\mathbb{A}|$ and $|\mathbb{B}|$ and apply Lemma **??** yielding a formula $\psi \in \mathrm{C}^k$ such that $\mathbb{A} \models \psi$ and $\mathbb{B} \models \neg\psi$. By Theorem 6.11, spoiler wins the bijective k-pebble game on $\mathbb{A}$ and $\mathbb{B}$. ∎

The Cai-Fürer-Immerman theorem

Because IFP over the counting sort can be simulated in P (since the counting sort has the same size as the input structure) we have the following.

Proposition 10.23. *Let φ be a formula of IFP+C. There is an algorithm which given a finite structure $\mathbb{A}$ decides whether it satisfies φ in time polynomial in $|\mathbb{A}|$.*

Many natural properties of relational structures can be expressed by the logic IFP+C. For instance, this logic can count the size of a maximal matching in a graph[1]. The Cai-Fürer-Immerman theorem proves that IFP+C does not contain all of P.

Theorem 10.24 (Cai-Fürer-Immerman, 1992)**.** *There is a class of structures which is in P but is not definable in IFP+C.*

The CFI construction

The CFI construction takes a cubic graph G and replaces each vertex by one of two gadgets, producing a hypergraph G' as follows.

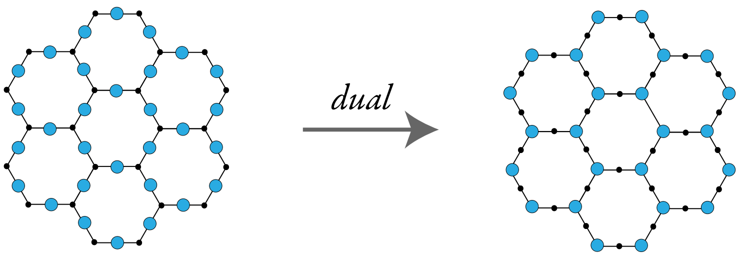
A hypergraph is a set of vertices V and a set E of hyperedges $e \subseteq V$. If $v \in e$ then we may say that v and e are *incident*. A graph is a special case of a hypergraph in which every hyperedge has exactly two elements. We will also consider 3-regular hypergraphs, that is hypergraphs where every hyperedge has exactly three elements.

If $\mathcal{H}$ and $\mathcal{H}'$ are hypergraphs then a *homomorphism* $f\colon \mathcal{H} \to \mathcal{H}'$ is a function mapping the vertices of $\mathcal{H}$ to vertices of $\mathcal{H}'$ such that for every hyperedge e of $\mathcal{H}$, its image $f(e)$ is a hyperedge in $\mathcal{H}'$ of the same cardinality (that is, f is $1-1$ on e).

[1]Anderson, Dawar, Holm, 2013

The *dual* of a hypergraph (V, E) is the hypergraph obtained by treating edges as vertices and vice-versa. More precisely, it has vertices E and the hyperedges are sets of the form $\{w \mid w \in e\}$, for $e \in E$.

Example 10.25. A cubic graph is a graph in which every vertex is incident to three edges. The dual of a cubic graph is a 3-regular hypergraph in which every vertex belongs to two hyperedges. In the picture below, vertices are depicted by black dots and hyperedges by blue dots.



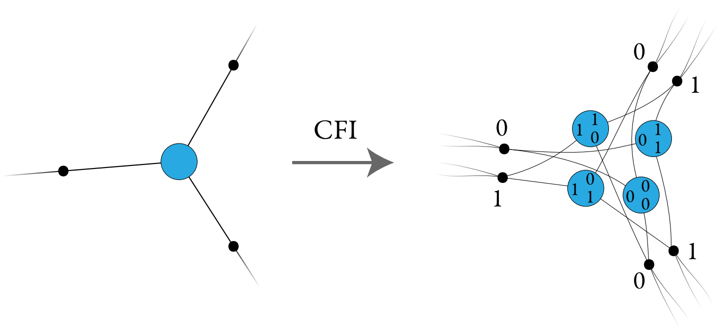


Figure 10.1: The even CFI gadget.

The CFI construction takes a hypergraph $\mathcal{H} = (V, E)$ together with a set $X \subseteq E$ of hyperedges and produces a hypergraph obtained by performing the following replacements. Firstly, every vertex v is replaced by a pair of vertices denoted

v_0 and v_1. Let $V \times \{0,1\}$ denote the vertices $\{v_0, v_1 \mid v \in V\}$ of $\mathcal{H}^X$. Note that this does not depend on X.

Next, every hyperedge $e = \{v^1, \dots, v^k\}$ is replaced by a gadget consisting of 2^{k-1} hyperedges. There are two possible gadgets, the *even gadget* and the *odd gadget*, and e is replaced by the odd gadget if $e \in X$ and by the even gadget otherwise. The even gadget (cf. Fig. 10.1) consists of all 2^{k-1} hyperedges of the form

$$\{v^1_{i_1}, \dots, v^k_{i_k}\} \qquad \text{where } i_1, \dots, i_k \in \{0,1\}, \qquad i_1 + \cdots + i_k \text{ is even}$$

and for the odd gadget, the sum should be odd.

Example 10.26. The simple case when $\mathcal{H}$ is a (graph) cycle with n vertices is illustrated in Fig. 10.2. The resulting (hyper)graph is either a disjoint union of

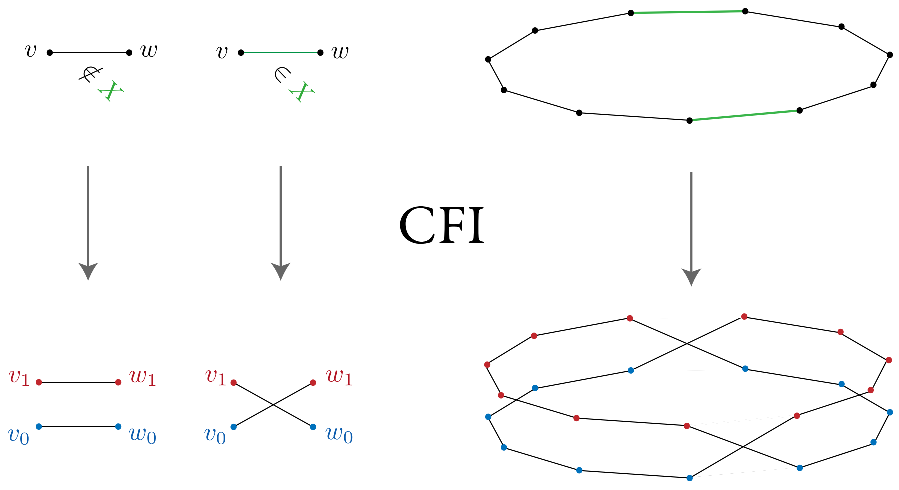


Figure 10.2: The CFI construction in the case of a cycle.

two cycles of length n if $|X|$ is even or a cycle of length $2n$ if $|X|$ is odd.

The class

From now on, we will consider hypergraphs of the form $\mathcal{H}^X$ where $\mathcal{H}$ is the dual of a connected cubic graph. We view such a hypergraph as a relational structure over the signature with a ternary relation T consisting of all triples (a, b, c) which form a hyperedge, and a binary relation $\sim$ denoting the equivalence relation which relates v_0 with v_1, for all $v \in V$.

Let $\mathcal{C}$ denote the class of structures isomorphic to $\mathcal{H}^\varnothing$ for some $\mathcal{H}$ which is the dual of a connected cubic graph. The following proposition immediately yields Theorem 10.24.

Proposition 10.27. *The class $\mathcal{C}$ is recognizable in polynomial time, and is not definable by any formula of IFP+C.*

The proof of the proposition is split into two parts, corresponding to its two statements.

Polynomial time algorithm

In what follows, we fix a hypergraph $\mathcal{H} = (V, E)$ which is the dual of a connected cubic graph. In particular, $\mathcal{H}$ is 3-regular, connected, and every vertex belongs precisely to two edges. Here connectedness means that for any pair of hyperedges e, f there is a path π connecting e and f.

We first prove the following:

Lemma 10.28. *For $\mathcal{H} = (V, E)$ as above and $X, Y \subseteq E$, the structures $\mathcal{H}^X$ and $\mathcal{H}^Y$ are isomorphic if and only if $|X| = |Y| \mod 2$.*

Proof. The key property of the two gadgets is that for any fixed $i \in \{1, \ldots, k\}$, swapping v_0^i with v_1^i in a gadget (that is replacing one by the other in each hyperedge) has the same effect as replacing the even gadget with the odd gadget, and vice-versa. This is formalized as follows. For a vertex $v \in V$ let $\hat{v}$ be the permutation of $V \times \{0, 1\}$ which is the transposition swapping v_0 and v_1. By the definition of the gadgets, we get:

Claim 1. Let $X \subseteq E$ and $v \in V$ be incident to two hyperedges, $e, f \in E$. Then the permutation $\hat{v}$ is an isomorphism between $\mathcal{H}^X$ and $\mathcal{H}^{X \triangle \{e,f\}}$.

A path π in $\mathcal{H}$ is a sequence

$$e_0, v_1, e_1, \ldots, v_{n-1}, e_n$$

where $e_0, \ldots, e_n$ are hyperedges, $v_1, \ldots, v_{n-1}$ are vertices such that v_i belongs to e_{i-1} and e_i. For such a path π, let $\hat{\pi}$ be the permutation of $V \times \{0,1\}$ which is the composition of all the transpositions $\hat{v}_i$, for $i = 1, \ldots, n-1$. This does not depend on order of composing the transpositions. Equivalently, $\hat{\pi}$ is the composition of all transpositions $\hat{v}$ for $v \in V$ with an odd number of occurrences in π.

From the previous claim we get:

Claim 2. Let $X \subseteq E$ and π be a path with endpoints $e, f \in E$. Then the permutation $\hat{\pi}$ is an isomorphism between $\mathcal{H}^X$ and $\mathcal{H}^{X \triangle \{e,f\}}$.

Finally, this yields:

Claim 3. Let $X, Y \subseteq E$ satisfy $|X| = |Y| \mod 2$. Then $\mathcal{H}^X$ and $\mathcal{H}^Y$ are isomorphic.

Proof. Note that $X \triangle Y$ has even size. Let $e_1, f_1, \ldots, e_k, f_k$ be all of its elements. For $i = 1, \ldots, k$ pick a path π_i joining e_i with f_i. By Claim 2, the composition $\hat{\pi}$ of all the permutations $\hat{\pi}_i$, for $i = 1, \ldots, k$ (in any order) is an isomorphism from $\mathcal{H}^X$ to $\mathcal{H}^Y$. ∎

To prove Lemma 10.28 it remains to show that if $\mathcal{H}^X$ is isomorphic to $\mathcal{H}^\emptyset$ then X has even size.

Let $\mathcal{H} = (V, E)$ be as above and $X \subseteq E$ and let $p \colon V \times \{0,1\} \to V$ be the projection homomorphism from $\mathcal{H}^X$ to $\mathcal{H}$.

A *solution* to $\mathcal{H}^X$ is a homomorphism $s \colon V \to V \times \{0,1\}$ which is a right-inverse of the projection p. Equivalently, as V corresponds to the set of equivalence classes of $\sim$ on $V \times \{0,1\}$, a solution to $\mathcal{H}^X$ is a choice function s picking one element $s(c) \in c$ from each $\sim$-equivalence class c in such a way that if three equivalence classes c_1, c_2, c_3 span a hyperedge in $\mathcal{H}^X$, then $\{s(c_1), s(c_2), s(c_3)\}$ form a hyperedge in $\mathcal{H}$.

More explicitly, s maps each vertex v of $\mathcal{H}$ to one of the vertices v_0, v_1 in such a way that for every hyperedge $\{u, v, w\}$ of $\mathcal{H}$, the vertices $s(u) = u_i, s(v) =$

$v_j, s(w) = w_j$ form a hyperedge in $\mathcal{H}$, that is, $i + j + k$ is even if $\{u, v, w\} \notin X$ and odd otherwise. Restated one last time, a solution s satisfies the following equations, for each hyperedge $e = \{a, b, c\} \in E$:

$$s(a) + s(b) + s(c) = [e \in X] \qquad \text{mod } 2, \tag{10.1}$$

where $[e \in X]$ is 1 if $e \in X$ and 0 otherwise.

Note that $\mathcal{H}^{\emptyset}$ has a solution: the mapping s which maps v to v_0, for all $v \in V$, is a solution. Hence, if X is even then $\mathcal{H}^X$ has a solution, as $\mathcal{H}^X$ is isomorphic to $\mathcal{H}^{\emptyset}$. Conversely, if a solution exists then $|X|$ is even – adding up the equations (10.1) for all $e \in E$ we get:

$$\sum_{\{a,b,c\} \in E} s(a) + s(b) + s(c) = \sum_{e \in E} [e \in X] = \sum_{e \in X} 1 = |X| \quad \text{mod } 2,$$

and the left-hand side is equal to $2 \sum_{v \in V} s(v)$ as every vertex belongs to exactly two hyperedges. In particular, if $|X|$ is odd then $\mathcal{H}^X$ is not isomorphic to $\mathcal{H}^{\emptyset}$. This proves Lemma 10.28. ∎

We now proceed to the proof of the first part of Proposition 10.27, namely, that $\mathcal{C}$ is recongizable in polynomial time.

Lemma 10.29. *The class $\mathcal{C}$ is recognizable in polynomial time.*

Proof. Given a structure $(W, T, \sim)$ first check that $\sim$ is an equivalence relation with equivalence classes of size 2, and that T is a symmetric, antireflexive relation. Hence, T defines a 3-regular hypergraph $\mathcal{G}$ on W.

Next, check that the quotient structure $\mathcal{G}/\sim$ is a connected 3-regular hypergraph where every vertex is incident with two hyperedges. Here, the quotient of the hypergraph $\mathcal{G}$ is the hypergraph whose elements are the equivalence classes of $\sim$ and where the hyperedges are formed by equivalence classes of vertices which form a hyperedge in $\mathcal{G}$.

Let $V = W/\sim$. For each equivalence class $v \in V$ arbitrarily denote by v_0 and v_1 its two elements. Next, check that for each hyperedge $\{u, v, w\}$ of $\mathcal{H}$ there is some $x \in \{0, 1\}$ such that for $i, j, k \in \{0, 1\}$, the hypergraph $\mathcal{G}$ contains the hyperedge $\{u_i, v_j, w_k\}$ if and only if $i + j + k = x \mod 2$. Let X be the set of all hyperedges $\{u, v, w\}$ as above for which $x = 1$.

By construction, $\mathcal{G}$ is isomorphic to $\mathcal{H}^X$, via the isomorphism mapping $v_i \in \mathcal{G}$ to $v_i \in \mathcal{H}^X$. In particular, by Lemma 10.28, $\mathcal{G} \in \mathcal{C}$ if and only if $|X|$ is even.

All this yields a polynomial time algorithm recognizing the class $\mathcal{C}$. ■

This completes the proof of the first part of Proposition 10.27.

Undefinability in IFP+C

We now show that $\mathcal{C}$ is not definable by any formula of IFP+C.

Proposition 10.30. *For every $k \in \mathbb{N}$ there is hypergraph $\mathcal{H}$ which is the dual of a connected cubic graph such that duplicator wins the k-pebble game on the structures $\mathcal{H}^\emptyset$ and $\mathcal{H}^{\{e\}}$ for any hyperedge e of $\mathcal{H}$.*

We first show how Proposition 10.30 proves Proposition 10.27.

Proof of Proposition 10.27. The first part is by Lemma 10.29. We show that $\mathcal{C}$ is not definable by any sentence of IFP+C, proving the proposition.

Suppose φ is a sentence of IFP+C defining $\mathcal{C}$. Let $k \in \mathbb{N}$ be as in Lemma 10.21. Then pick $\mathcal{H}$ as in Proposition 10.30. By Corollary 10.22, $\mathcal{H}^{\{e\}} \in \mathcal{C}$. This contradicts Lemma 10.28. ■

Consider the graph which delineates the $n \times n$ hexagonal tessellation depicted in Fig. 10.3 – each hexagon is surrounded by six vertices and six edges, and the graph consists of vertices and edges.

We view such a graph as a graph on the torus by identifying the leftmost vertices with the rightmost ones, and the vertices at the top with the vertices at the bottom. This gives a cubic graph with $2n^2$ vertices, denoted T_n.

The only properties of the graphs T_n that we use are that they are cubic, of increasing size, and for large n, do not have small edge separators splitting them into two big parts, as formalised in the following lemma. Appropriate tessellations of spheres or other surfaces would also work.

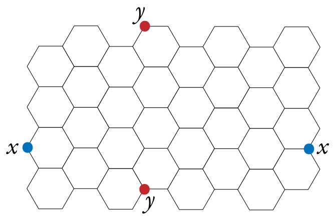


Figure 10.3: A 8×8 hexagonal tesselation of the torus – each column of vertices has 8 vertices, and there are 8 columns. Vertices on the boundary are identified as indicated.

Lemma 10.31. *Fix a number k, and consider a set F of at most k edges in T_n, where n is large enough. Then there is a connected component of $T_n - F$ of size at least $2n^2 - ck^2$, for some constant $c > 0$ (independent from k, n).*

If n is large enough comparing to k, there is at most one such component, so in this case we can call it the *big component* of $T_n - F$.

Lemma 10.31 follows from a variant of the isoperimetric inequality, for the hexagonal tessellation of the plane: any closed path of length k can enclose at most $O(k^2)$ vertices. This inequality can be proved combinatorially, but it also follows from the classical isoperimetric inequality.

Proof of Proposition 10.30. Fix $k \in \mathbb{N}$. Let n be large enough, as specified by Lemma 10.31. Let $\mathcal{H} = (V, E)$ be the hypergraph dual to T_n. Then for every $F \subseteq V$ with $|F| \leqslant k$ we have that $\mathcal{H} - F$ has a unique connected component with more than half the vertices.

Let $e_0 \in E$. We show that duplicator wins the bijective k-pebble game on $\mathcal{H}^{\emptyset}$ and $\mathcal{H}^{\{e_0\}}$. Let $\bar{y}$ be the set of pebble names, with $|\bar{y}| = k$.

The intuition behind the proof below is as follows. Recall that if $e \neq e_0$ then $\mathcal{H}^{\{e_0,e\}}$ is isomorphic to $\mathcal{H}^{\emptyset}$, where the isomorphism is given by $\hat{\pi}$, for any path π joining e_0 with e. Hence, in the game on $\mathcal{H}^{\emptyset}$ and $\mathcal{H}^{\{e_0,e\}}$, duplicator wins by always playing $\hat{\pi}$.

Duplicator will behave as if the structure $\mathcal{H}^{\{e_0\}}$ was actually $\mathcal{H}^{\{e_0,e\}}$ for some hyperedge e which is in the big connected component of $\mathcal{H} - F$, where F is the set of projections of the pebbled positions. Duplicator will maintain a path π connecting e_0 and e. The hyperedge e will be moved further away once spoiler places his pebbles near e, and the path π will be updated accordingly. The bijection played by duplicator is $\hat{\pi}$.

More precisely, as the pebble game will be played, duplicator will maintain the invariant that at any time, the current position $(\bar{a}, \bar{b})$ with $\bar{a} \in (V \times \{0,1\})^{\bar{y}}$ and $\bar{b} \in (V \times \{0,1\})^{\bar{y}}$ is such that $p(\bar{a}) = p(\bar{b})$, that is, if a pebble is placed on a vertex v_i in $\mathcal{H}^{\varnothing}$ then the corresponding pebble in $\mathcal{H}^{\{e_0\}}$ is either placed on v_i or v_{1-i}. Let $\bar{s} = p(\bar{a}) = p(\bar{b}) \in V^{\bar{y}}$; those are the vertices v in V such that v_0 or v_1 is pebbled. Duplicator will also maintain a hyperedge e in the big component of $\mathcal{H} - \bar{s}$, and a path π joining e_0 with e, such that $\hat{\pi}(\bar{a}) = \bar{b}$. Initially, $\bar{a}$ and $\bar{b}$ are empty, $e = e_0$ and π is the trivial path starting and ending at e_0.

Suppose at position $(\bar{a}, \bar{b})$ of the game spoiler declares he will move one of his pebbles $x \in \bar{y}$ to a new position. Let $\bar{s} = p(\bar{a}) = p(\bar{b})$. Duplicator picks a hyperedge e' in the big component of $\mathcal{H} - \bar{s}$ which is not incident to any vertex in $\bar{s}$. Note that e and e' are in the same connected component of $\mathcal{H} - \bar{s}$. The new path π' is obtained from π by appending a path connecting e and e' in $\mathcal{H} - \bar{s}$. Duplicator responds with the bijection $\hat{\pi}$.

Now spoiler places the pebble x. Suppose the pebble is placed on a vertex w_0 in structure $\mathcal{H}^{\varnothing}$ (the case of a vertex w_1 is similar). So in the structure $\mathcal{H}^{\{e_0\}}$ the pebble is placed on $w_j = \hat{\pi}'(w_0)$, where j is 0 or 1 depending on the parity of the number of occurrences of w in π'. By definition, the invariant $\hat{\pi}'(\bar{a}[x/w_0]) = \bar{b}[x/w_j]$ is maintained. It remains to check that this response of duplicator is not a loosing move.

Suppose for example that the vertices with pebbles x, y, z form a hyperedge f in $\mathcal{H}^{\varnothing}$, and to focus attention, suppose $f = \{w_0, u_0, v_0\}$ for some $u, v \in V$. We need to check that $\hat{\pi}'(f)$ is a hyperedge in $\mathcal{H}^{\{e_0\}}$. As u and v belong to the tuple $\bar{s}$ and e' is not adjacent to $\bar{s}$, we have that $f \neq e'$. As π' has endpoints e_0 and e', in total the vertices u, v, w are visited by π' an even number of times, unless $f = e_0$. Then $\hat{\pi}'(f) = \{w_i, u_j, v_k\}$ where $i + j + k$ is even if $f \neq e_0$ and odd if

$f = e_0$. In either case, $\hat{\pi}'(f)$ is a hyperedge in $\mathcal{H}^{\{e_0\}}$, as required.

This handles the case when f is a hyperedge of the form $\{w_0, u_0, v_0\}$. The general case, when $f = \{w_i, u_j, v_k\}$ forms/does not form a hyperedge in $\mathcal{H}^{\emptyset}$, proceeds similarly.

This shows that duplicator's response $\hat{\pi}'$ is not loosing, and maintains the invariant. Hence, duplicator wins the bijective k-pebble game. ■

This finishes the proof of Theorem 10.24.

Corollary 10.32. *There is a class of graphs which is in* P *but is not definable in IFP+C.*

Proof. This can be done by interpreting the class $\mathcal{C}$ in finite graphs. In this particular case, the interpretation can be simplfied slightly, yielding the class of bipartite graphs which are duals of the hypergraphs in $\mathcal{C}$. In particular, those graphs are bipartite, all vertices in one part have degree 3, and all vertices in the other part have degree 4, and there are further conditions needed to express that the graph stems from $\mathcal{H}^{\emptyset}$ for the dual $\mathcal{H}^{\emptyset}$ of a connected graph $\mathcal{H}$. ■

Corollary 10.33. *For every fixed $k \in \mathbb{N}$, there are two non-isomorphic graphs G, H which are not distinguished by k-WL.*

10.4 *Graph isomorphism, canonisation, and capturing polynomial time*

Fix a signature Σ; we will consider mostly the signature of graphs below. Below by a *logic* $\mathcal{L}$ we mean a computable language over some fixed alphabet (the *syntax*), together with a binary relation (the semantics) $\models$ between the class of all Σ-structures and $\mathcal{L}$.

Let $\mathcal{L}$ be a logic and C be a complexity class. Say that $\mathcal{L}$ *captures* C on a class of Σ-structures $\mathcal{C}$ if the following conditions hold:

- for any property $P \in$ C of Σ-structures there is a formula $\varphi \in \mathcal{L}$ such that for all $\mathbb{A} \in \mathcal{C}$,

$$\mathbb{A} \in P \quad \iff \quad \mathbb{A} \models \varphi,$$

- conversely, for every formula $\varphi \in \mathcal{L}$ the class $\{\mathbb{A} \mid \mathbb{A} \in \mathcal{C}, \mathbb{A} \models \varphi\}$ is *effectively* in C.

The effectivity condition above means that there is an algorithm which given φ produces a Turing machine which recognizes $\{\mathbb{A} \mid \mathbb{A} \in \mathcal{C}, \mathbb{A} \models \varphi\}$ and whose language is in C. Without the effectiveness condition, we could have degenerate logics as the following one. Fix any enumeration $P_1, P_2, \ldots$ of all the properties of structures that are in the class C. The logic $\mathcal{L}$ consists of all natural numbers, and $\mathbb{A} \models i$ if $\mathbb{A} \in P_i$. This would satisfy the second condition above, without the effectivity condition.

The existential second-order logic captures NP, by Fagin's theorem. By the Immerman-Vardi theorem (Thm. 10.12), IFP captures P on the class of all finite, ordered graphs. The central problem in descriptive complexity is:

is there a logic which captures P *on the class of all finite graphs?*

By the CFI theorem (cf. Cor. 10.32), the logic IFP+C does not capture P on the class of all finite graphs, as it fails to satisfy the first part of the above definition. A negative answer to the above question would imply P $\neq$ NP, due to Fagin's theorem. On the other hand, a positive answer – especially one with a "natural" logic – might provide some insight into polynomial time computation.

Apart from classes of ordered structures, there are classes for which a logic capturing P is known. In particular, IFP+C captures P on the following classes of graphs:

- the class of all trees,
- the class of all planar graphs,
- any class excluding a fixed minor.

Definable orders

Suppose there is a formula $\varphi(x, y)$ of IFP which defines a total order on any graph G from a fixed class $\mathcal{C}$. Then the Immerman-Vardi theorem implies that IFP captures polynomial time on $\mathcal{C}$.

However, often to define an order one needs to fix several parameters. For example, if G is a path of nonzero length then there is no formula $\varphi(x, y)$ which defines an order on G, due to the nontrivial automorphism of G which swaps the endpoints. However, if an endpoint is fixed, then we can define an order using IFP on G. Say that a formula $\varphi(\bar{z}, x, y)$ *defines orders* on a class $\mathcal{C}$ if for every $G \in \mathcal{C}$ there is a tuple $\bar{c} \in G^{\bar{z}}$ such that $\varphi(\bar{c}, x, y)$ defines an order on G. For example, there is a formula $\varphi(z, x, y)$ of IFP which defines orders on the class $\mathcal{C}$ of all paths.

Proposition 10.34. *If $\varphi(\bar{z}, x, y)$ is a formula of IFP which defines orders on $\mathcal{C}$ then IFP captures* P *on $\mathcal{C}$.*

Proof. Let P be a polynomial-time property and let ψ be a sentence of IFP defining P, obtained from the Immerman-Vardi theorem.

Define a new sentence ψ' which existentially quantifies over the variables $\bar{z}$, checks that $\varphi(\bar{z}, x, y)$ defines a linear order on the given structure, and then evaluates ψ with $x < y$ interpreted as $\psi(\bar{z}, x, y)$. It follows by construction that ψ' holds in a graph $G \in \mathcal{C}$ if and only if $G \in P$. ∎

Corollary 10.35. *IFP captures* P *on the class of all finite paths, and on the class of all cycles.*

Exercise 10.36. Prove that IFP captures P on the class of all graphs of maximum degree 2.

The above method has serious limitations, however: there is no formula (of any logic) which defines linear orders on the class of all edgeless graphs, due to the symmetries of those graphs. For this reason, we will relax the notion of defining an order on G itself, and will require producing an order on an isomorphic copy of G instead. We formalize this below.

Graph canonisation

A graph canonisation algorithm for a class $\mathcal{C}$ is an algorithm which given a graph $G \in \mathcal{C}$ outputs an ordered graph G' with vertices $\{0, \ldots, |G| - 1\}$ which is isomorphic to G, in such a way that if G and H are isomorphic graphs then G' and H' are equal graphs. For example, there is a polynomial-time canonisation for the class of cliques, and there is a polynomial-time canonisation algorithm for the class of stars (trees of depth 2).

If $\mathcal{C}$ has a polynomial-time canonisation algorithm, then there is a logic which captures P on $\mathcal{C}$. This logic is IFP over the signature $\{E, \leqslant\}$, and a graph $G \in \mathcal{C}$ satisfies a sentence φ if G' satisfies φ, where G' is the ordered copy of G produced by the canonisation algorithm and $\leqslant$ is interpreted as the usual order on $\{0, \ldots, |G| - 1\}$. This logic captures P on $\mathcal{C}$ by the Immerman-Vardi theorem, but is not very natural, at it refers to an external algorithm.

A more natural logic capturing P can be obtained if the canonisation can be produced by a formula. Say that $\mathcal{C}$ has *IFP+C-definable canonisation* if there are formulas of IFP+C

- $\varphi_{\mathrm{OK}}(\bar{z})$, defining the correct choices of the parameters $\bar{z}$,
- $\varphi_E(\bar{z}, x, y)$, where x and y range over the number sort (recall that IFP+C formulas may produce numbers $\{0, \ldots, n-1\}$ in a structure with n elements),

such that for every $G \in \mathcal{C}$ there are $\bar{c} \in G^{\bar{z}}$ such that $G \models \varphi_{\mathrm{OK}}(\bar{c})$ and the graph with vertices $\{0, \ldots, |G| - 1\}$ (the elements of the number sort) and edges defined by $\varphi_E(\bar{c}, x, y)$, is isomorphic to G.

Say that a class $\mathcal{C}$ of graphs has *Weisfeiler-Leman dimension k* if k-WL identifies every graph $G \in \mathcal{C}$.

Example 10.37. The class of forests has Weisfeiler-Leman dimension 2. Also, it has IFP+C definable canonisation.

Proposition 10.38. *Suppose $\mathcal{C}$ has IFP+C definable canonisation. Then:*

1. *$\mathcal{C}$ has a polynomial-time canonisation algorithm,*
2. *IFP+C captures P on $\mathcal{C}$,*
3. *$\mathcal{C}$ has finite Weisfeiler-Leman dimension.*

Proof. (1). Recall that a fixed formula $\varphi(\bar{z}, x, y)$ of IFP+C can be evaluated in polynomial time on a given graph G. In particular, given a graph G, we can scan all tuples $\bar{c} \in G^{\bar{z}}$ and find the first one which satisfies $\varphi_{\text{OK}}(\bar{z})$. Then compute the edge relation defined by $\varphi_E(\bar{c}, x, y)$ on $\{0, \ldots, |G| - 1\}$ and output the resulting graph.

(2) follows from the Immerman-Vardi theorem (Thm. 10.12) just like in the proof of Proposition 10.34.

(3). Let $\varphi(\bar{x}, y, z)$ be the formula defining the canonization on $\mathcal{C}$. As in Lemma 10.21, for the formula φ we can get a number $k \in \mathbb{N}$ such that for every two graphs G, H and tuples $\bar{a} \in G^{\bar{z}}$ and $\bar{b} \in H^{\bar{z}}$, if $(G, \bar{a}) \equiv_{C^k} (H, \bar{b})$ then $|G| = |H|$ and for all $i, j \in \{0, \ldots, |G| - 1\}$, $G \models \varphi(\bar{a}, i, j)$ if and only if $H \models \varphi(\bar{b}, i, j)$. Hence, by definition of φ, if $G \in \mathcal{C}$ and H is a graph such that $G \equiv_{C^k} H$ then $G \cong H$. By Corollary 6.12, k-WL distinguishes every $G \in \mathcal{C}$, so $\mathcal{C}$ has Weisfeiler-Leman dimension at most k. ■

It can be proved, often with great effort, that the following classes of graphs admit IFP+C definable canonisation:

- the class of forests,
- the class of planar graphs,
- any class of graphs excluding a fixed minor,
- the class of graphs of bounded cliquewidth.

Proving results like this is usually much more involved than proving the existence of a polynomial-time canonisation algorithm, and requires a very fine understanding of the combinatorial structure of the considered graphs. By Proposition 10.38, this proves that IFP+C captures P on each of those classes.

11 Outlook

In this final chapter, we review some of the currently active areas of research in finite model theory. This list is by far not complete.

Model checking

One area is the pursuit of finding classes of graphs (or other structures) for which model checking is fixed-parameter tractable for some logic (essentially, that means that a sentence φ can be tested on a given graph $G \in \mathcal{C}$ in time $f(\varphi, \mathcal{C}) \cdot |G|^c$ for some function f and constant c). We have seen that MSO is fixed-parameter on any class of graphs with bounded treewidth, or more generally, bounded cliquewidth. Similarly, we have seen that model checking FO is fixed-parameter tractable on any class of graphs with locally bounded treewidth. This direction of research has been pushed to its limits on graph classes which are closed under taking subgraphs. It is known that for such graph classes, model checking FO is fixed-parameter tractable if and only if the class if *nowhere dense*. An active direction of research is to obtain results of this form for graph classes which are not closed under taking subgraphs. A first example of a positive result in this direction is for classes of bounded cliquewidth, for which model checking MSO is fixed-parameter tractable. This result has been very recently extended to some classes of *bounded twin-width*, for which model checking FO is fixed-parameter tractable.

Satisfiability

A direction which has not been discussed at all in these lectures is finding fragments of first-order logic (or similar logics) for which the satisfiability problem is decidable (since it is undecidable for full first-order logic, cf. Thm. 4.2). Such fragments include various variants of first-order logic with two variables, or *guarded logics*.

Definable canonisation

Another active direction of research in finite model theory is to find classes of structures for which canonisation can be carried out in some logic. Examples include graphs of bounded rankwidth, for which admit IFP+C definable canonisation, and graphs of bounded treewidth, which admit some weak form of MSO-definable canonisation.

Capturing polynomial time

The quest for finding a logic which captures polynomial time computation currently consists of two main directions. The first direction tries to extend the logic IFP+C by various additional mechanisms apart from counting, such as the possibility of solving systems of equations over finite fields. This allows in particular to solve some variants of the CFI query which cannot be solved by IFP+C. Although it seems unlikely that such logics capture P, proving this requires a better understanding of the CFI construction, and producing more and more elaborate versions of it. The second main direction is focussed on another candidate logic for capturing P, described below.

*Choiceless Polynomial Time**

Choiceless polynomial time ($\tilde{\mathrm{C}}$PT), and its extension with counting ($\tilde{\mathrm{C}}$PT+C) look much like a programming language with bounded resources. It is an open problem whether $\tilde{\mathrm{C}}$PT captures polynomial time computation (on unordered graphs).

A rough description of $\tilde{\text{C}}$PT is given below. Essentially, $\tilde{\text{C}}$PT is a variant of while-programs, where:

- the basic type is a *hereditarily finite set with atoms*. The *atoms* are the elements of the input structure. A hereditarily finite set with atoms is either an atom or a finite set of hereditarily finite sets with atoms (defined recursively). For example $\{\{a,b\},\{b,c\},\{a,c\}\}$ is a hereditarily finite set with atoms, if a,b,c are atoms.

- the programming language has variables which can store hereditarily finite sets with atoms. Expressions also evaluate to hereditarily finite sets with atoms.

- an input graph $G=(V,E)$ is represented by setting the variables V and E to the set of vertices and the set of edges (a set of two-element sets of atoms),

- the programming language is equipped with assignment instructions $\mathtt{v} := e$, where $\mathtt{v}$ is a variable and e is an expression (see below), sequential composition of instructions $I_1; I_2$, the **while** loop and the **if else** conditional, where a condition is an expression, interpreted as true if it evaluates to a nonempty set. Finally, there is a **return** instruction which terminates the computation and returns a output value.

- expressions can be manipulated using basic set-theoretic constructions such as $\cup, \cap$ and $-$, as well as $\bigcup$ (the set union of a family), the expression $\mathbf{unique}(x)$ returing the unique element of a singleton x (or $\emptyset$ if it does not exist) and the *set comprehension* construct

  $$\{f(\mathtt{x}) \mid \mathtt{x} \in e\},$$

  where f is an expression with variable $\mathtt{x}$ and e is an expression.

- in the counting extension, $\#X$ is an expression which evaluates to the cardinality of a set X, represented using the von Neumann encoding (with $0=\emptyset, 1=\{0\}, 2=\{0,1\}, \ldots, n=\{0,\ldots,n-1\}$).

*omitted in the lectures

To bound the resources, it is required that there is a polynomial p such that for every input graph G with n vertices, the program terminates after at most $p(n)$ computation steps, and moreover, the computation involves at most $p(n)$ hereditarily finite sets in total (as sets stored in the variables, or in their elements, or their elements, etc.). Slightly more precisely, say that a hereditarily finite set X *occurrs* in a state of the program if there is some variable which stores a hereditarily finite set Y such that $X = Y$, or X belongs to an element of Y, or an element of an element of Y, etc. Then the requirement is that there are at most $p(n)$ hereditarily finite sets which occur in some state of the program, during its computation on input G.

A $\tilde{\text{C}}$PT program is then a pair (I, p) where I is an instruction as described above, and $p\colon \mathbb{N} \to \mathbb{N}$ bounding its running time and its resources (if the instruction exceeds the time or resources specified by p on some input, it terminates and returns $\emptyset$). It is not difficult to show that every such program can be evaluated in polynomial time, so that $\tilde{\text{C}}$PT$\subseteq$ P and similarly $\tilde{\text{C}}$PT+C$\subseteq$ P. The converse inclusion $\tilde{\text{C}}$PT+C$\subseteq$ P is wide open. As $\tilde{\text{C}}$PT is quite like a programming language and less like a logic, it seems quite difficult to prove inexpressibility results for $\tilde{\text{C}}$PT, even for problems which do not belong to P.

It is known, although nontrivial, that it can solve a variant of the CFI query (namely the variant in which the resulting hypergraphs are equipped with a total quasi-order whose equivalence classes are the sets $\{v_0, v_1\}$ of size 2). There are ongoing efforts to push this result further and capture more complicated variants of the CFI query or special cases of the isomorphism problem for graphs equipped with a total quasi-order whose equivalence classes have bounded size.

A

Automata on Words and Trees

In this appendix we will develop the finite model theory of words and trees using automata. It turns out that monadic second-order logic behaves particularly well on those structures, due to their good decomposability properties. Roughly, a word or tree can be decomposed into two disjoint parts which behave in an independent way. This is related to the usefuleness of automata on words and trees. We will see that the set of words satisfying a given first-order sentence is a regular language. In fact, sets of words definable by finite automata correspond precisely to sets which are definable in monadic second-order logic. This correspondence lifts to the case of trees. Hence, automata provide a powerful tool for studying the expressive power of monadic second-order logic on words and trees. From this, we will derive decidability of the satisfiability problem, as well as tractability of the model checking problem on words and trees.

A.1 Monadic second-order logic

In this section we will see that over the class of finite trees, the satisfiability problem for first-order logic is decidable. This is even the case for the much more powerful monadic second-order logic.

A.2 Words

Monadic second-order logic is important because of its connection to automata and regular languages. A finite word w over an alphabet A may be viewed as a finite structure, as depicted in Fig. A.1. Its elements correspond to the *positions*

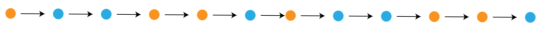


Figure A.1: A word as a structure.

of w, that is, the set $\{1, \ldots, n\}$, where n is the length of w. For each letter of A there is a unary symbol that marks the positions carrying the letter. There is a binary relation *succ* representing the successor relation on the positions. Let Σ_A denote the resulting signature.

For a sentence φ let $L(\varphi)$ denote the set of words $w \in A^*$ which satisfy φ, when viewed as a Σ_A-structure.

Example A.1. The following formula expresses that the sets X and Y partition the positions in an alternating way:

$$\mathit{alternate}(X,Y) \equiv \forall x.[x \in X \leftrightarrow x \notin Y] \wedge \forall x,y.[\mathit{succ}(x,y) \rightarrow (x \in X \leftrightarrow y \in Y)]$$

The following sentence expresses that the word has even length:

$$\exists X,Y.\mathit{alternate}(X,Y) \wedge \forall x.[\neg\exists y.\mathit{succ}(y,x) \rightarrow (x \in X)] \wedge [\neg\exists y.\mathit{succ}(x,y) \rightarrow (x \in Y)].$$

Note that the sentence above has the form

$$\exists X_1 \ldots \exists X_n.\varphi,$$

where φ does not use second-order quantifiers. A formula of this form is called an *existential* second-order formula.

It is not difficult to generalize the example above to express an arbitrary regular language.

Lemma A.2. *For every regular language $L \subseteq A^*$ there is a sentence φ of existential monadic second-order logic such that $L(\varphi) = L$.*

Proof. We show that for every regular language $L \subseteq A^*$ there is a sentence φ of monadic second-order logic such that $L(\varphi) = L$. Let $\mathcal{A}$ be a nondeterministic finite automaton recognizing L. Let Q be its set of states. We ignore the empty word in the reasoning, as it can be treated separately by writing a sentence detecting that the domain is empty.

The idea is that a run $q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} \dots \xrightarrow{a_n} q_n$ of $\mathcal{A}$ over a nonempty word $w = a_1 \cdots a_n$ can be represented by a partition of the positions of w into $|Q|$ sets, X_q for each $q \in Q$, where X_q contains a position $i \in \{1, \dots, n\}$ if $q_i = q$ (see Fig. A.5). Partitions corresponding to accepting runs are characterized as

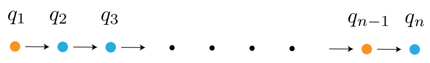


Figure A.2: A run of an automaton on a word.

follows:

- for every successive positions x and y, if $x \in X_p, y \in X_q$ and y carries the label a, then $p \xrightarrow{a} q$ is a transition of $\mathcal{A}$;
- if the first position carries a label $a \in A$ then it belongs to some set X_q such that $p \xrightarrow{a} q$ is a transition of $\mathcal{A}$ for some initial state p;
- the last position y belongs to X_q for some accepting state q.

Accepting runs of $\mathcal{A}$ (with the first state removed) correspond bijectively to partitions X_q as above. Note that the three conditions above can be expressed by a formula ρ with free variables $(X_q)_{q \in Q}$ using only first-order quantification.

Write a sentence φ starting with a sequence of existential quantifiers $\exists X_q$, one for each $q \in Q$, followed by a conjunction of a formula expressing that $(X_q)_{q \in Q}$ partitions the vertices with the formula ρ above. Then $w \in L$ if and only if $w \in L(\varphi)$. ∎

We will show the converse to Lemma A.2 – that each sentence of monadic second-order logic defines a regular language of words. Roughly speaking, monadic second-order logic is built out of a few atomic relations using Boolean combinations and existential quantification. On the other hand, regular languages are closed under Boolean combinations and *projections*, which correspond to existential quantification as follows.

A word $w \in A^*$ together with a set U of positions of w can be seen as a word $w \otimes U$ over the alphabet $A \times \{0,1\}$, where the second component indicates whether the position is in U. Suppose we have a formula $\psi = \exists X.\varphi(X)$ and we have inductively shown that the set L of words $w \otimes U \in (A \times \{0,1\})^*$ such that $w \models \varphi(U)$ is regular. Then the set of words $w \in A^*$ which satisfy ψ is obtained from $L \subseteq (A \times \{0,1\})^*$ by projecting out the second component of each letter.

More generally, for two alphabets B and A, a function $\pi\colon B \to A$ and a language $L \subseteq B^*$, let $\pi(L) \subseteq A^*$ be the language of all words $\pi(b_1) \cdots \pi(b_k)$ such that $b_1 \cdots b_k \in L$. Say that $\pi(L)$ is the *projection* of L along π.

Lemma A.3. *Regular languages are closed under union, complementation, and projections.*

Proof. The cases of unions and complements are well known (for complements, the automaton is first determinised using the powerset construction). We treat the case of projections. Let $\mathcal{A}$ be a nondeterministic finite automaton recognizing $L \subseteq B^*$ and let $\pi\colon B \to A$. Define an automaton $\mathcal{A}'$ with the same states as $\mathcal{A}$, the same initial states, the same accepting states, and where each transition $p \xrightarrow{b} q$ is replaced by $p \xrightarrow{\pi(b)} q$. It is easy to check that $L(\mathcal{A}') = \pi(L(\mathcal{A}))$. ∎

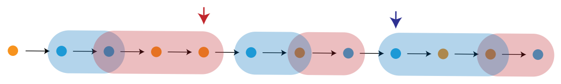
Theorem A.4. *Fix an alphabet A and let Σ_A be the associated signature. The following conditions are equivalent for a language $L \subseteq A^*$:*

- *$L = L(\mathcal{A})$ for some nondeterministic finite automaton $\mathcal{A}$ over the alphabet A,*
- *$L = L(\varphi)$ for some Σ_A-sentence φ of monadic second-order logic.*

Both translations are effective.

Proof. The downward implication is by Lemma A.2. For the upward implication, we prove a stronger statement concerning formulas φ with free second-order variables.

Fix a formula φ with free variables $\mathcal{V}$ which are either first-order variables or second-order variables. A valuation v of $\mathcal{V}$ in a word w defines a word $w \otimes v$ over the extended alphabet $A \times \{0,1\}^{\mathcal{V}}$, as depicted in Fig. A.3.



● a ● b	a	b	b	a	a	b	a	b	b	a	a	b
● Y	0	1	1	0	0	1	1	0	1	1	1	0
● X	0	0	1	1	1	0	1	1	0	0	1	1
$\rightarrow$ y	0	0	0	0	1	0	0	0	0	0	0	0
$\rightarrow$ x	0	0	0	0	0	0	0	0	1	0	0	0

Figure A.3: Viewing a structure with a valuation as a word over an extended alphabet.

Define the language $L(\varphi)$ of φ to be the set of all words $w \otimes v$ such that $w \in A^*$ and v is a valuation of the free variables $\mathcal{V}$ of φ in the structure $\mathbb{A}$ associated to w, such that v satisfies φ in $\mathbb{A}$. By induction on φ, we show that $L(\varphi) \subseteq (A \times \{0,1\}^{\mathcal{V}})^*$ is regular.

In the base case, we need to handle atomic formulas: $x = y$, $x \in X$, $succ(x,y)$ and $a(x)$ for each $a \in A$. For each case it is easy to construct an automaton or regular expression defining $L(\varphi)$. For instance, if φ is $a(x)$ then the regular expression is $K^* \cdot (a,1) \cdot K^*$ where K is the set of all letters of the form $(b,0)$, for $b \in A$.

In the inductive step, we need to handle Boolean combinations and first- and second-order existential quantification. The case of Boolean combinations follows, since regular languages are closed under Boolean combinations.

We treat the second-order existential quantifier; the proof in the first-order case is the same. Suppose $\varphi = \exists X.\psi$ for some formula ψ with free variables $\mathcal{V} \cup \{X\}$. Let $\mathcal{A}$ be a nondeterministic automaton recognizing $L(\psi) \subseteq (A \times \{0,1\}^{\mathcal{V}\cup\{X\}})^*$, obtained by the inductive assumption. Then $L(\varphi)$ is the projection of $L(\psi)$ obtained by removing the X component of each letter. As regular languages are closed under projections, this case follows.

This ends the inductive proof that $L(\varphi)$ is regular, for every formula φ. The statement of the theorem is obtained as a special case. ■

Corollary A.5. *It is decidable whether a given sentence φ of monadic second-order logic is satisfied in some finite word.*

Proof. Construct an automaton $\mathcal{A}$ recognizing $L(\varphi)$, and test its emptiness. ■

A.3 Ranked trees

The above results for finite words can be lifted to finite trees. It is easiest to define automata for *ranked trees*, that is, trees with a fixed bound on the number of children of each node. The case of unranked trees can be reduced to the case of ranked trees (in fact, binary trees), as done in the next section.

Let A be a finite set of labels, called the *alphabet*. Assume that each label $a \in A$ has a specified *rank*, which is a natural number. We then call A a *ranked alphabet*. A ranked tree over A is a tree in which a node with label a of rank r has exactly r children, numbered from 1 to r (cf. Fig. A.4). For a nonempty ranked finite tree to exist, at least one letter must have rank 0.

A ranked tree t over a ranked alphabet A can be seen as a logical structure as follows:

- for each $a \in A$ there is a unary relation a, marking the nodes with label a;
- there are binary relations $child_i$, for each i from 1 to the maximal rank of a label in A. The relation $child_i$ relates a node v with a node w if v is the ith child of w (it doesn't need to exist).

Denote by Σ_A the resulting signature. We can then speak about properties of ranked trees over A which are definable in monadic second-order logic over the

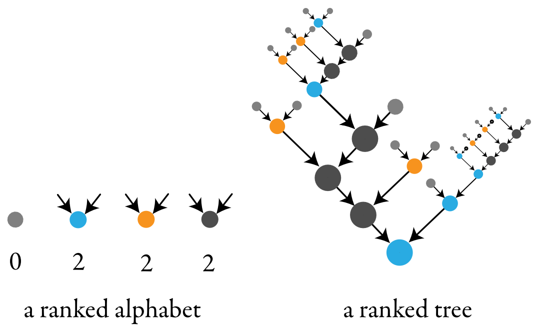


Figure A.4: A ranked alphabet and a ranked tree.

signature Σ_A. The goal now is to define a notion of automata which recognize precisely all such properties.

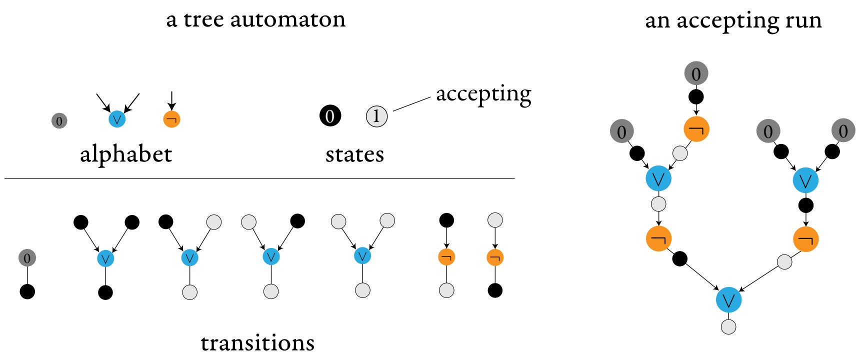
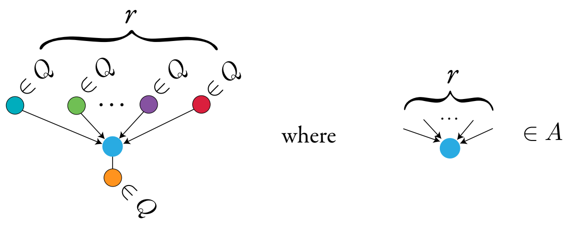


Figure A.5: A tree automaton computing the value of a Boolean expression using $\vee, \neg$ and 0, and its run.

A *nondeterministic tree automaton* $\mathcal{A}$ over a ranked alphabet A consists of:

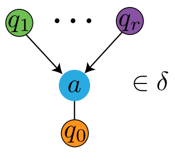
- a finite set of states Q,

- a set δ of transitions of the form:



- a set of accepting states $F \subseteq Q$.

A run of $\mathcal{A}$ on a ranked tree t (cf. Fig. A.5) is a function ρ mapping the nodes of t to Q such that for every node v with label a and r children, if ρ maps v to q_0 and the children of v to $q_1, \ldots, q_r$, respectively, then



A run is *accepting* if it maps the root to an accepting state. By $L(\mathcal{A})$ we denote the set of all ranked trees t for which there is an accepting run of $\mathcal{A}$. A set L of ranked trees over a ranked alphabet A is called *regular* if it is equal to $L(\mathcal{A})$ for some tree automaton $\mathcal{A}$.

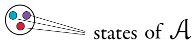
The following fact is proved similarly as in the case of words:

Lemma A.6. *The class of regular tree languages is closed under Boolean combinations.*

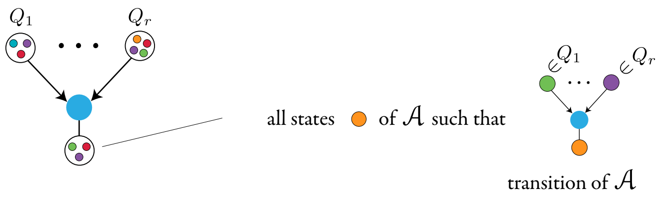
Proof. Let $\mathcal{A}$ and $\mathcal{B}$ be two tree automata over a ranked alphabet A. Define the *disjoint union* of $\mathcal{A}$ and $\mathcal{B}$ as the automaton $\mathcal{A} \uplus \mathcal{B}$ whose set of states is the disjoint union of the states of $\mathcal{A}$ and $\mathcal{B}$, and analogously for the transitions and the accepting states. Then $L(\mathcal{A} \uplus \mathcal{B}) = L(\mathcal{A}) \cup L(\mathcal{B})$.

To show that regular tree languages are closed under complementation, we employ the usual powerset construction. For a tree automaton $\mathcal{A}$ we construct a (deterministic, in a suitable sense) automaton with the following components:

- the states are all sets of states of $\mathcal{A}$. Each state can be depicted as



- the transitions are


Q_1
Q_r
all states ● of $\mathcal{A}$ such that
$\in Q_1$
$\in Q_r$
transition of $\mathcal{A}$

- the accepting states are the states which do not contain any accepting state of $\mathcal{A}$.

It is easy to check that this automaton recognizes the complement of $L(\mathcal{A})$. ■

Let A and B be two ranked alphabets and let $\pi \colon A \to B$ be a rank-preserving function, that is, the rank of $\pi(a)$ is equal to the rank of a. Then every ranked tree t over A yields a ranked tree $\pi(t)$ over B, obtained by replacing the labels according to π. For a set of ranked trees L over A, let $\pi(L) = \{\pi(t) \mid t \in L\}$.

Lemma A.7. *If L is a regular tree language then $\pi(L)$ is regular, too.*

Proof. Let $\mathcal{A}$ be a tree automaton with $L(\mathcal{A}) = L$. Define a tree automaton $\mathcal{B}$ with the same states as $\mathcal{A}$, the same accepting states, and the same transitions, but where the alphabet letter in each transition is relabelled according to π. Then $L(\mathcal{B}) = \pi(L)$. ■

We then prove:

Theorem A.8. *Fix a ranked alphabet A and let Σ_A be the associated signature. The following conditions are equivalent for a set L of ranked trees:*

- *$L = L(\mathcal{A})$ for some tree automaton $\mathcal{A}$ over the ranked alphabet A,*

- *$L = L(\varphi)$ for some Σ_A-sentence φ of monadic second-order logic.*

Both translations are effective.

Proof. We follow the same proof as in Theorem A.4.

For the downwards implication, as in Lemma A.2 note that an accepting run of a tree automaton $\mathcal{A}$ on a tree t can be represented by a partition of the nodes into a family of sets X_q, one per each state $q \in Q$ of $\mathcal{A}$. Moreover, a formula ψ with free variables $(X_q)_{q \in Q}$ can express that given sets form a partition which corresponds to an accepting run of $\mathcal{A}$. We take φ to be ψ preceded by a sequence of second-order existential quantifiers $\exists X_q$, for each $q \in Q$.

For the upwards implication, again we prove a stronger statement by induction on a formula φ, possibly with free variables $\mathcal{V}$. For such a formula define a ranked alphabet $A \times \{0, 1\}^{\mathcal{V}}$, where the rank of a letter is the rank of its first component in A. A ranked tree t over A together with a valuation v of the free variables $\mathcal{V}$ defines a ranked tree $t \otimes v$ over $A \times \{0, 1\}^{\mathcal{V}}$. Define the language $L(\varphi)$ as the set of all ranked trees $t \otimes v$ where t is a ranked tree over A and v is a valuation of $\mathcal{V}$ in t which satisfies φ in t.

We show that $L(\varphi)$ is regular by induction on φ. The base case amounts to constructing automata for each of the atomic predicates $x = y$, $x \in X$, $child_i(x, y)$, and $a(x)$ for $a \in A$. Each case is easily solved by hand.

In the inductive step, we use Lemma A.6 to handle Boolean combinations and Lemma A.7 to handle existential quantification. ∎

Lemma A.9. *There is an algorithm which, given a tree automaton $\mathcal{A}$ and a tree t decides in time $\mathcal{O}(|\mathcal{A}| \cdot |t|)$ whether $\mathcal{A}$ accepts t.*

Proof. Given t, run $\mathcal{A}$ on t, from leaves to the root. At each node v, store the set $X_v \subseteq Q$ of all states $q \in Q$ such that there is a run on the subtree of v with q being the state at v. The set X_v can be computed in time $\mathcal{O}(|\mathcal{A}|)$, given the sets $X_{v_1}, \ldots, X_{v_k}$ computed recursively at all the children $v_1, \ldots, v_k$ of v. Altogether, this gives an algorithm with the specified running time. ∎

We now show that emptiness of tree automata is decidable, in polynomial time.

Lemma A.10. *There is an algorithm which given a tree automaton $\mathcal{A}$ decides in polynomial time whether $L(\mathcal{A})$ is empty.*

Proof. The *depth* of a run on a tree t is the maximal number of nodes on a root-to-leaf path in t. The *root state* of a run is the state it assigns to the root. $L(\mathcal{A})$ is nonempty if and only if there is some run whose root state is accepting.

The algorithm computes the set R_ω of states q of $\mathcal{A}$ such that $\mathcal{A}$ has some run with root state q.

For $i = 1, 2, \ldots$, let R_i be the set of states q of $\mathcal{A}$ such that $\mathcal{A}$ has some run with root state q of depth at most i.

Then

$$R_1 \subseteq R_2 \subseteq \ldots \subseteq Q.$$

R_1 can be easily computed from the transition relation δ of $\mathcal{A}$.

Take an accepting run ρ of depth at most $i+1$ and suppose that the root has label a of rank $r > 0$. Then ρ decomposes into r runs of depth at most i. Hence, for all $i \geqslant 1$,

$$R_{i+1} = \{q_{r+1} \in Q \mid (q_1, \ldots, q_{r+1}) \in \delta_a \text{ for some } a \text{ of rank } r \text{ and } q_1, \ldots, q_r \in R_i\}.$$

In particular, the set R_{i+1} can be computed in polynomial time, given R_i. As the sequence $R_1 \subseteq R_2 \subseteq \ldots$ is increasing, after at most $|Q|$ steps, it must stabilize. Suppose that $R_i = R_{i+1}$. It follows that $R_i = R_{i+1} = R_{i+2} = \ldots = R_\omega$ is the set of states q of $\mathcal{A}$ such that $\mathcal{A}$ has some run with root state q.

By the above, R_ω can be computed in polynomial time. Finally, $L(\mathcal{A}) = \emptyset$ if and only if $R_\omega \cap F = \emptyset$. ∎

Corollary A.11. *Fix a ranked alphabet A. It is decidable whether a given sentence φ of monadic second-order logic is satisfied in some finite ranked tree over A.*

Proof. Convert φ to an automaton $\mathcal{A}$ using Theorem A.8. Next, test emptiness of $\mathcal{A}$ using Lemma A.10. ∎

Corollary A.12. *Fix a sentence φ of monadic second-order logic. There is an algorithm which given a ranked tree t, decides in time $\mathcal{O}(|t|)$ whether t satisfies φ.*

Proof. Follows from Lemma A.9. ∎

A.4 Unranked trees

Fix a finite alphabet A. A *rooted, labelled tree* is a directed graph (possibly empty) in which every node is labelled by an element of A, as depicted below:

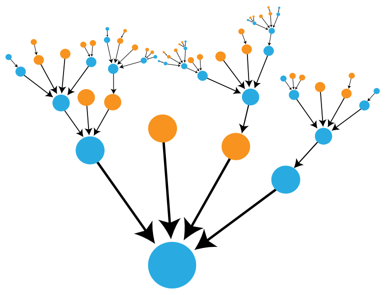


Figure A.6: A rooted, labelled, unranked tree.

Such a tree can be naturally viewed as a structure over the signature Σ_A consisting of the binary *parent* relation, as well as unary predicates for each label $a \in A$.

Our goal is to prove the following generalization of Corollary A.11 to unranked trees.

Theorem A.13. *It is decidable whether a given sentence φ of monadic second-order logic is satisfied in some finite labelled tree.*

To prove Theorem 8.10, we still need to address the case of unranked trees. To do so, encode unranked trees in ranked trees, as depicted below.

We now describe the details of the construction.

Proof of Theorem 8.10. Let A be an unranked alphabet. Construct a ranked alphabet $A' = \{a' \mid a \in A\} \cup \{B, E\}$ where a' is a copy of $a \in A$ and has rank 2, B has rank 2, and E has rank 0. Define a transformation T mapping an unranked tree t with labels from an alphabet A to a ranked tree $T(t)$ over the ranked alphabet

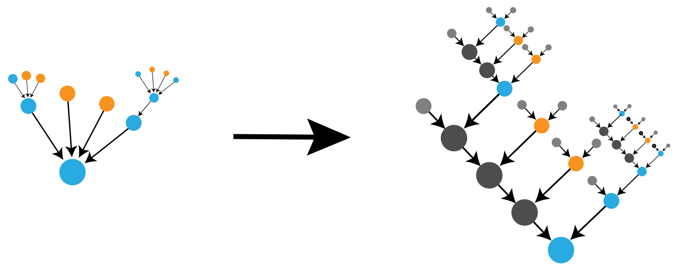


Figure A.7: Transforming an unranked tree to a ranked tree.

A', by induction as follows. If t has a root v with label a and children whose subtrees are $t_1, \ldots, t_r$, then $T(t)$ has a path of $r+1$ nodes $v_0, v_1, \ldots, v_r$, with labels $a', B, B, \ldots, B, E$, respectively, where each subsequent node is the first child of the previous one. The nodes $v_0, \ldots, v_{r-1}$ each have a second child which is the root of the ranked tree $T(t_i)$ constructed recursively (cf. Fig A.7).

The tree t can be decoded from $T(t)$ by contracting every node with label E or B with its parent. The key point is that this decoding can be carried out by formulas of monadic second-order logic, as follows.

In any ranked tree t' over the alphabet A', define a unary relation *node*, unary relations a for each $a \in A$, and a binary relation *parent*, such that:

- *node* holds at the nodes with label a', for some $a \in A$;
- the predicate $a \in A$ holds at the nodes with label a';
- *parent* holds between two nodes v, w, if and only if $v, w \in$ *node*, and w is an ancestor of v in t', and every node on the path from w to v (exclusively) is labelled by B.

Clearly, the relations *node* and *parent* above are definable by formulas of monadic second-order logic which are independent of t'. Denote those formulas *node*$'$ and *parent*$'$.

Let $I(t')$ denote the A-labelled structure with elements consisting of all nodes satisfying *node*, additionally equipped with the relations a for $a \in A$ and *parent* as defined above.

Claim 4. For every unranked tree t over the alphabet A, the structure $I(T(t))$ is a labelled tree isomorphic to t.

Now, there is a sentence γ which holds in a ranked tree t' if and only if it is isomorphic to a tree $T(t)$, for some t: γ says that a node is labelled with B or E if and only if it is a first child. From the previous claim, we get:

Claim 5. Every unranked tree t over the alphabet A is isomorphic to $I(t')$ for some ranked tree t' over the alphabet A' which satisfies γ.

We now complete the proof of Theorem 8.10. Let φ be any sentence of monadic second-order logic on A-labelled unranked trees. Translate φ on A-labelled unranked trees to a sentence φ' on A'-labelled ranked trees, by substituting each of the atomic predicates $a(x)$ and $child(x,y)$ by the formulas obtained above. For example:

$$\varphi \equiv \forall x \exists y. a(x) \to (parent(x,y) \wedge b(y))$$

is transformed into

$$\varphi' \equiv \forall x. node'(x) \to [\exists y. node'(y) \ \wedge (a'(x) \to parent'(x,y) \wedge b'(y))].$$

In this way we obtain a sentence φ' such that for all t':

t' *satisfies* φ' if and only if $I(t')$ *satisfies* φ.

It follows from Claim 5 that the following conditions are equivalent:

- some unranked tree over the alphabet A satisfies φ,
- some ranked tree over the alphabet A' satisfies $\gamma \wedge \varphi'$.

By Corollary A.11, the latter problem is decidable. This proves Theorem 8.10, as $\gamma \wedge \varphi'$ can be effectively obtained, given φ. ■

Rabin's celebrated result extends Theorem 4.2 to the class of countable trees. The general outline of the proof is again similar as in the case of finite words or trees, however, significant obstacles need to be overcome. First, the definition of automata needs to be modified substantially, as infinite trees lack leaves which allowed to kick off the computation in the finite case. Second, proving closure under complementation is rather involved.

Apart from obtaining an algorithm for satisfiability on unranked trees, we also get a linear-time algorithm for model checking.

Corollary A.14. *Let φ be a sentence of monadic second-order logic. There is an algorithm which given a tree t decides in time $\mathcal{O}_{\varphi}(|t|)$ whether t satisfies φ.*

Proof. We use the notation from the proof of Theorem 8.10. The additional observation is that the ranked tree $T(t)$ can be computed from t in linear time. Moreover, t satisfies φ if and only if $T(t)$ satisfies φ'. Apply Corollary A.12 to φ' and $T(t)$. ■

B
Interpretations

Let $\mathcal{C}$ and $\mathcal{D}$ be two classes of structures, over signatures Σ and Γ, respectively. Assume for simplicity that Σ is relational. Let $I\colon \mathcal{D} \rightharpoonup \mathcal{C}$ be a partial map from $\mathcal{D}$ to $\mathcal{C}$ such that:

- the domain of I is definable by a Γ-sentence γ. That means that $I(\mathbb{D})$ is defined if and only if $\mathbb{D}$ satisfies γ.
- every structure in $\mathcal{C}$ is isomorphic to some structure in the image of I.
- for each $\mathbb{D} \in \mathrm{Dom}(I)$, the structure $I(\mathbb{D})$ is definable by formulas independent of $\mathbb{D}$. More precisely, there is a Γ-formula $\delta(x)$ and Γ-formulas $R'(x_1, \ldots, x_r)$, for each $R \in \Sigma$ of arity r, with the following property. For each $\mathbb{D} \in \mathcal{D}$, the domain of $I(\mathbb{D})$ is $\delta_{\mathbb{D}}$, and for each relation $R \in \Sigma$ of arity r, the relation $R_{I(\mathcal{D})}$ is equal to the restriction of $R'_{\mathbb{D}}$ to $\delta_{\mathbb{D}}$.

Call an operation as above an *$\mathcal{L}$-interpretation* of $\mathcal{C}$ in $\mathcal{D}$ if all the Γ-formulas above belong to a logic $\mathcal{L}$. If such an interpretation exists, then say that $\mathcal{C}$ *interprets in* $\mathcal{D}$, or that $\mathcal{D}$ *interprets* $\mathcal{C}$, via an $\mathcal{L}$-interpretation.

In the previous section, we showed that the class of unranked trees over an alphabet A interprets in the class of ranked trees over the alphabet A' via a monadic second-order interpretation. The argument finishing the proof of Theorem 8.10 generalizes to:

Theorem B.1. *Let $\mathcal{L}$ be first-order logic or monadic second-order logic. If $\mathcal{C}$ interprets in $\mathcal{D}$ via an $\mathcal{L}$-interpretation $I\colon \mathcal{D} \rightharpoonup \mathcal{C}$ then the satisfiability problem for the logic $\mathcal{L}$ over the class $\mathcal{C}$ reduces to the satisfiability problem over the class $\mathcal{D}$.*

Proof. For each Σ-formula φ of monadic second-order logic define a Γ-formula φ' by induction as follows:

$$\begin{aligned} \exists x.\alpha &\mapsto \exists x.\delta(x) \wedge \alpha' & R(x_1, \dots, x_r) &\mapsto R'(x_1, \dots, x_r) \\ \exists X.\alpha &\mapsto \exists X.\alpha' & x = y &\mapsto x = y \\ \alpha \vee \beta &\mapsto \alpha' \vee \beta' & x \in X &\mapsto x \in X \\ \neg\alpha &\mapsto \neg\alpha' & \bot &\mapsto \bot \end{aligned}$$

Note that this translation maps first-order formulas to first-order formulas.

Let $\mathbb{D} \in \mathrm{Dom}(I)$ and φ be a Σ-formula with free variables $\mathcal{V}$. Let v be a valuation of $\mathcal{V}$ in the set $\delta_{\mathbb{D}}$. The following equivalence is shown by induction on φ:

$$I(\mathbb{D}), v \models \varphi \quad \Leftrightarrow \quad \mathbb{D}, v \models \varphi'.$$

In particular, if φ is a sentence, φ holds in $I(\mathbb{D})$ if and only if φ' holds in $\mathbb{D}$. Hence, φ is satisfiable in $\mathcal{C}$ if and only if $\varphi' \wedge \gamma$ is satisfiable in $\mathcal{D}$. ■

The proof shows that the model-checking problem over $\mathcal{C}$ can be reduced to the model-checking problem over $\mathcal{D}$ if we assume that the second condition in the definition of an interpretation is efficient.

Corollary B.2. *Suppose $I\colon \mathcal{D} \rightharpoonup \mathcal{C}$ is an $\mathcal{L}$-interpretation such that for every structure $\mathbb{C} \in \mathcal{C}$ one can compute in time $p(|\mathbb{C}|)$ a structure $\mathbb{D} \in \mathcal{D}$ such that $I(\mathbb{D})$ is isomorphic to $\mathbb{C}$. Then there is a reduction of the model-checking problem for $\mathcal{L}$ on $\mathbb{C} \in \mathcal{C}$ to the model-checking problem for $\mathcal{L}$ on $\mathcal{D}$ which, given $\mathbb{C} \in \mathcal{C}$ and $\varphi \in \mathcal{L}$ outputs in time $p(|\mathbb{C}|) + poly(\varphi)$ a structure $\mathbb{D} \in \mathcal{D}$ and $\varphi' \in \mathcal{L}$ such that*

$$\mathbb{C} \models \varphi \quad \Leftrightarrow \quad \mathbb{D} \models \varphi'.$$

It is often useful to compose interpretations.

Lemma B.3. *Let $\mathcal{L}$ be first-order logic or monadic second-order logic. If $I\colon \mathcal{D} \rightharpoonup \mathcal{C}$ and $J\colon \mathcal{C} \rightharpoonup \mathcal{B}$ are $\mathcal{L}$-interpretations, then the composition $J \circ I\colon \mathcal{D} \rightharpoonup \mathcal{B}$ is an $\mathcal{L}$-interpretation.*

B.1 Cliquewidth and MSO-interpretations of trees

The following theorem states that every class of bounded cliquewidth interprets in a class of trees, via an interpretation of monadic second-order logic.

Theorem B.4. *Fix $k \in \mathbb{N}$ and let C be a set with k elements. There is an interpretation I of monadic second-order logic such that $I(t) = [t]$ for each clique decomposition t with colors C. In particular, the class $\mathcal{C}_k$ of graphs of cliquewidth k interprets in a class of trees over a fixed alphabet, via an interpretation of monadic second-order logic.*

Proof. First, for each $i \in C$ define a formula $\chi_i(x)$ such that for a clique decomposition t with colors C and leaf a of t

$$t \models \chi_i(a) \quad \Longleftrightarrow \quad \textit{a has color i in [t].}$$

The formula $\chi_i(x)$ works as follows: it existentially quantifies over sets X_i, for each $i \in C$, and asserts that:

- $(X_i)_{i \in C}$ form a partition of the ancestors of x,
- if x is labeled c_i then $x \in X_i$,
- for every ancestor $v \in X_i$ of x and its parent v' (if it exists), if v' is a node of rank 1 labelled $\mathit{recolor}_f$, then $v' \in X_{f(i)}$; otherwise, if v' is a node of rank 2 then $v' \in X_i$.

For a node c of t, let $t|_c$ denote the subtree of t rooted at c. Write a binary formula $\chi'_i(x, z)$ such that for a clique decomposition t with colors C and leaf a of t and its ancestor c,

$$t \models \chi'_i(a, c) \quad \Longleftrightarrow \quad \textit{a has color i in } [t|_c].$$

The formula $\chi'_i(x, z)$ is just $\alpha_i(x)$ with all quantifiers constrained to quantify over vertices which are descendants of z.

Finally, we define a formula $\varphi(x, y)$ such that for any clique decomposition t with colors C and leaves a, b of t,

$$t \models \varphi(a, b) \quad \Longleftrightarrow \quad \textit{a, b are adjacent in [t].}$$

The formula $\varphi(x, y)$ expresses that for some $M \subseteq \{\{i, j\} \mid i, j \in C\}$ and $\{i, j\} \in M$, the following hold:

- the greatest common ancestor z of x and y is labelled $join_M$,
- $\chi'_i(x, z)$,
- $\chi'_i(y, z)$.

The formulas $\chi_i(x)$ together with $\varphi(x, y)$ yield the required interpretation I. ■

Corollary B.5. *Satisfiability of monadic second-order logic is decidable over $\mathcal{C}_k$.*

Corollary B.6. *There is an algorithm which inputs a sentence φ of monadic second-order logic and a graph G together with its clique decomposition t of width k, and decides if G satisfies φ in time $\mathcal{O}_{k,\varphi}(|t|)$.*

Theorem B.7. *Let $\mathcal{T}$ be a class of trees and let $\mathcal{C}$ be a class that interprets in $\mathcal{T}$ via an interpretation of monadic second-order logic. Then $\mathcal{C}$ has bounded cliquewidth.*

Interpretations of trees

Theorem B.4 proves that the class of graphs of cliquewidth at most k interprets in a class of ranked trees.

In this section we prove a converse: if $\mathcal{C}$ is a class of graphs which interprets in a class of ranked trees via an interpretation of monadic second-order logic, then $\mathcal{C}$ has bounded cliquewidth.

Fix a ranked alphabet A. Let $\varphi(x, y)$ be a formula of monadic second-order logic. For a ranked tree t, let $\varphi(t)$ denote the graph G whose vertices are the leaves of t, and where two vertices a, b are adjacent in G if and only if $t \models \varphi(a, b)$.

Lemma B.8. *For any ranked tree t, the graph $\varphi(t)$ has cliquewidth bounded by a constant depending only φ.*

Proof. Let $\mathcal{A}$ be a deterministic tree automaton corresponding to $\varphi(x, y)$ as described in Theorem A.8. The automaton $\mathcal{A}$ is over the alphabet $A \times \{x, y\}$, and let Q be its set of states.

Given a ranked tree t over the alphabet A and a node a, let $\mathcal{A}(t \otimes [x \mapsto a]) \in Q$ denote the root state of the automaton in its run on the tree t extended by the valuation mapping x to a and leaving y undefined. Similarly, define $\mathcal{A}(t \otimes [y \mapsto a]) \in Q$.

A node c of t defines a function $\gamma\colon Q \to Q$, such that running the automaton $\mathcal{A}$ on the tree t with the subtree rooted at c replaced by a single leaf with state q yields the state $\gamma(q)$ at the root of t. Denote the function γ by $\delta_{t\setminus c}$.

Then, for two leaves a, b of t, the state $\mathcal{A}(t \otimes [x \mapsto a, y \mapsto b])$ depends only on the following:

- the state $p = \mathcal{A}(t \otimes [x \mapsto a]) \in Q$
- the state $q = \mathcal{A}(t \otimes [y \mapsto b]) \in Q$
- the label λ of the least common ancestor c of a and b in t,
- the transition function $\delta_{t\setminus c}\colon Q \to Q$.

Let X consist of all tuples $(p, q, \lambda, \delta_{t\setminus c})$ such that $\mathcal{A}(t \otimes [x \mapsto a, y \mapsto b])$ is an accepting state of $\mathcal{A}$.

Define a set of colors $C = Q \cup (Q \times \{0, 1\})$. For every subtree t' of t we define a clique decomposition with colors C of the subgraph of $\varphi(t)$ induced by the leaves of t', with the coloring mapping a leaf a to $\mathcal{A}(t \otimes [x \mapsto a]) \in Q$.

The decomposition is constructed by induction on the size of t'. The base case, when t' has just one node, is trivial. In the inductive step, suppose t' has root labeled λ and two subtrees, t_0 and t_1, and let d_0 and d_1 be the two clique decompositions obtained by inductive assumption. Define a clique decomposition d as the following term:

$$d := \mathit{recolor}_f(\mathit{join}_S(\mathit{recolor}_{i_0}(d_0), \mathit{recolor}_{i_1}(d_1))),$$

where:

- $i_0\colon C \to C$ is any function such that $i_0(q) = (q, 0)$ for $q \in Q$;
- $i_1\colon C \to C$ is any function such that $i_1(q) = (q, 1)$ for $q \in Q$;

- S consists of all sets $\{p,q\}$ with $p,q \in Q$ such that $(p,q,\lambda,\delta_{t\backslash c}) \in X$, where λ is the label of the root c of t' and $\delta_{t\backslash c}\colon Q \to Q$ is the function defined by c in t;
- $f\colon C \to Q \subseteq C$ is any function such that $f((p,0),(q,1)) = f((p,1),(q,0))$ is the state r of $\mathcal{A}$ such that (p,q,λ,r) is a transition of $\mathcal{A}$.

By definition of the set X, the obtained clique decomposition d satisfies the required property. This ends the inductive definition.

In the end, we obtain a clique decomposition of $\varphi(t)$ of width $3|Q|$. ∎

From this, we get that classes of bounded cliquewidth are preserved by interpretations:

Theorem B.9. *Let $\mathcal{C}$ be a class of graphs of bounded cliquewidth and let I be an interpretation of monadic second-order logic. Then $I(\mathcal{C})$ is a class of bounded cliquewidth.*

Proof. Let $\mathcal{C}$ be a class of cliquewidth at most k. By Theorem B.4 there is an intepretation I_k such that $I_k(t) = [t]$ for each clique decomposition with k colors. The graph $I(I_k(t))$ is a graph of cliquewidth bounded only by k and $I \circ I_k$, by Lemma B.8. The class of all graphs $I(I_k(t))$ contains $I(\mathcal{C})$, so $I(\mathcal{C})$ is a class of bounded cliquewidth. ∎

Exercise B.10. Prove Lemma B.3.

Exercise B.11. Show that the class of grids expanded with finitely many unary predicates, as considered in the proof of Theorem 4.2, first-order interprets in the class $\mathcal{C}$ of subgraphs of planar (undirected) grid graphs. Conclude that satisfiability of first-order logic is undecidable over $\mathcal{C}$.

Exercise B.12. Show that the class of all finite graphs MSO-interprets in the class of subgraphs of planar grid graphs.

Exercise B.13. Show that for each finite relational signature Σ, the class of all finite Σ-structures interprets in the class of all finite graphs via a first-order interpretation. First interpret the class of finite Σ-structures in the class of finite labelled graphs, and then interpret the latter in the class of finite graphs.

Exercise B.14. Show that the class of all finite graphs interprets in the class of all finite partial orders.

Exercise B.15. Let $\mathcal{C}$ be the class consisting of all structures with domain $\{1,\ldots,n\} \times \{1,\ldots,m\}$, for all $m,n \in \mathbb{N}$, equipped with the binary relation $\sim_1$ relating elements with equal first coordinate, and the binary relation $\sim_2$ relating elements with equal second coordinate, and a unary predicate U which is an arbitrary subset of the domain. Prove that the class of finite graphs first-order interprets in $\mathcal{C}$.

Exercise B.16. A tree order is a partially ordered set $(X,\leqslant)$ such that for every $x \in X$, $\{y \mid y \leqslant x\}$ is totally ordered by $\leqslant$, and for every $x,y \in X$ there is z such that $z \leqslant x$ and $z \leqslant y$. For example, $(\{0,1\}^*,\preceq)$, where $\preceq$ is the prefix order, is a tree order, and every total order is a tree order.

Prove that $(\mathbb{Q},\leqslant)$ first-order interprets in $(\{0,1\}^*,\preceq,U_0)$, where $U_0 \subseteq \{0,1\}^*$ is the unary predicate consisting of words ending with 0.

Exercise B.17. Prove that the class of countable tree orders first-order interprets in the class of all structures $(\{0,1\}^\omega,\preceq,U_0,W)$, where $\{0,1\}^\omega$ is the set of infinite $0,1$-sequences, where W ranges over all possible subsets of $\{0,1\}^\omega$.

Bibliography