Computable Structure Theory: Part 1 Draft

Antonio Montalbán

Draft or Part 1 - Compiled on April 14, 2017

[0]Saved: April 13, 17
Compiled: April 14, 2017

Contents

Preface

We all know that in mathematics there are proofs that are more difficult than others, constructions that are more complicated than others, and objects that are harder to describe than others. The objective of *computable mathematics* is to study this complexity, to measure it, and to understand where it comes from. Among the many aspects of mathematical practice, in this book we concentrate on the complexity of structures. By *structures*, we mean objects like rings, fields, or linear orderings, which consist of a domain on which we have relations, functions, and constants.

Computable structure theory studies the interplay between complexity and structure. By *complexity*, we mean descriptional or computational complexity, in the sense of how difficult it is to describe or compute a certain object. By *structure*, we refer to algebraic or structural properties of mathematical structures. The setting is that of infinite countable structures and thus, within the whole hierarchy of complexity levels, the appropriate tools to measure complexity are those used in computability theory: Turing degrees, the arithmetic hierarchy, the hyperarithmetic hierarchy, etc. These structures are like the ones studied in model theory, and we will use a few basic tools from there too. The intention is not, however, to effectivize model theory. The motivations come from questions of the following sort: Are there syntactical properties that explain why certain objects (like structures, relations, isomorphisms, etc.) are easier or harder to compute or to describe?

The objective of this book is to describe some of the main ideas and techniques used in the field. Most of these ideas are old, but for many of them, the style of the presentation is not. Over the last few years, the author has developed new frameworks for dealing with these old ideas — as for instance for forcing, r.i.c.e. relations, jumps, Scott ranks, back-and-forth types, etc. One of the objectives of the book is to present these frameworks in a concise and self-contained form.

Of great influence to the modern state of the field, and also to the author's view of the subject, is the monograph by Ash and Knight [**AK00**] published in 2000. There is, of course, some intersection between that book and this one. But even on that intersection, the approach is different.

The intended readers are graduate students and researchers working on mathematical logic. Basic background in computability theory and logic, as is covered in standard undergraduate courses in logic and computability, is assumed. The objective of this book is to describe some of the main ideas and techniques of the field so that graduate students and researchers can use it for their own research.

The monograph will consist of three parts: introductory topics, transfinite topics, and classes of structures.

Part I, Introductory topics, is about the part of the theory that can be developed below a single Turing jump. The first chapters introduce what the author sees as the basic tools to develop the theory: ω-presentations, relations, and $\exists$-atomic structures, as treated by the author in [**Mon09, Mon12, Mon13c, Mona**]. Many of the topics covered in Part I (like Scott sentences, 1-generics, the method of true stages, categoricity, etc.) will then be generalized through the transfinite in part II.

Part II, Transfinite topics, is closer to what is covered in Ash and Knight's book [**AK00**]. The main chapters in Part II are those on forcing and the α-priority method. The exposition of forcing is only aesthetically new (it will be similar to that developed for [**HTMM**]). The presentation of Ash's α-priority method will be more than just aesthetically different. It will use the method of α-true stages developed in [**Mone**].

Part III, Classes of structures, is about the computability theoretic properties of classes of structures, rather than of single structures. The first chapter deals to connections with descriptive set theory, and it will contain some of the more recent work from [**Mon13a, Monb, MM**]. The chapter on comparability of classes treats old topics like Borel reducibility, but also newer topics like effective reducibility of classes of computable structures [**FF09, FFH$^+$12, Monc**] and the connections between functors and interpretability [**HTMMM, HTMM**]. Σ-small classes, covered in the last chapter, have been a recurrent topic in the author's work, as they touch on many aspects of the theory and help to explain previously observed behaviors [**HM12, HM, Mon10, Mon13b**].

Notation and Conventions

The intention of this section is to refresh the basic concepts of computability theory and structures and set up the basic notation we use throughout the book. If the reader has not seen computable functions before, this section may be too fast of an introduction, in which case we recommend starting with other textbooks like Cutland [**Cut80**], Cooper [**Coo04**], Enderton [**End11**], or Soare [**Soa87**].

The computable functions

A function $f\colon \mathbb{N} \to \mathbb{N}$ is *computable* if there is a computer program that, on input n, outputs $f(n)$. This might appear to be too informal a definition, but the Turing–Church thesis tells us that it does not matter which method of computation you choose, you always get the same class of functions from $\mathbb{N}$ to $\mathbb{N}$, as long as the method allows for enough basic functionality. The reader may choose to keep in mind whichever definition of computability feels intuitively more comfortable, be it Turing machines, μ-recursive functions, lambda calculus, register machines, Pascal, Basic, C++, Java, Haskel, or Python.[1] We will not use any particular definition of computability, and instead, every time we need to define a computable function, we will just describe the algorithm in English and let the reader convince himself or herself it can be written in the programing language he or she has in mind.

The choice to use $\mathbb{N}$ as the domain and image for the computable functions is not as restrictive as it may sound. All finite objects can be encoded using a single natural number. Even if formally we think of computable functions as having domain $\mathbb{N}$ for simplicity, we think of them as using any kind of finite object as inputs or outputs. This should not be surprising. It is what computers do when they encode everything you see on the screen using finite binary strings, or equivalently, natural numbers written in binary. For instance, we can encode pairs of natural numbers by a single number using the *Cantor paring function* $(x, y) \mapsto ((x+y)(x+y+1)+y)/2$, which is a bijection from $\mathbb{N}^2$ to $\mathbb{N}$ whose inverse is easily computable too. One can then encode triples by using pairs of pairs, and then encode n-tuples, and then tuples of arbitrary size, and then tuples of tuples, etc. The same way, we can consider standard effective bijections between $\mathbb{N}$ and various other sets like $\mathbb{Z}$, $\mathbb{Q}$, V_ω, $\mathcal{L}_{\omega,\omega}$, etc. Given any such finite object a, we use Quine's notation $\ulcorner a \urcorner$ to denote the number coding a. Which method of coding we use is inmaterial for us so long as the method is sufficiently effective. We will just assume these methods exist and hope the reader can figure out how to define them.

[1] For the reader with a computer science background, let us remark that we do not impose any time or space bound on our computations — computations just need to halt and return an answer after a finite amount of time using a finite amount of memory.

Let

$$\Phi_0, \Phi_1, \Phi_2, \Phi_3, ...$$

be an enumeration of the computer programs for functions from $\mathbb{N}$ to $\mathbb{N}$, say ordered alphabetically. Given n, we write $\Phi_e(n)$ for the output of the eth program on input n. Each Φ_e computes a partial function $\mathbb{N} \rightharpoonup \mathbb{N}$. Let us remark this function may not be total because, on some inputs, $\Phi_e(n)$ may run forever and never halt with an answer. We call these the *partial computable functions*. The *computable functions* are the total functions among the partial computable ones. We write $\Phi_e(n) \downarrow$ to mean that this computation *converges*, that is, that it halts after a finite number of steps; and we write $\Phi_e(n) \uparrow$ to mean that it *diverges*, i.e., does not halt and never returns an answer. Computers or even Turing machines run on a step-by-step basis. We use $\Phi_{e,s}(n)$ to denote the output of $\Phi_e(n)$ after s steps of computation, which can be either not converging yet ($\Phi_{e,s}(n) \uparrow$) or converging to a number ($\Phi_{e,s}(n) \downarrow = m$). Notice that, given e, s, n, we can decide whether $\Phi_{e,s}(n)$ converges or not, computably, as all we have to do is run $\Phi_e(n)$ for s steps. If f and g are partial functions, we write $f(n) = g(m)$ to mean that either both $f(n)$ and $g(m)$ are undefined, or both are defined and have the same value. We write $f = g$ if $f(n) = g(n)$ for all n. If $f(n) = \Phi_e(n)$ for all n, we say that e is an *index* for f. The *Padding Lemma* states that every program has infinitely many indices — just add dummy instructions at the end of a program, getting essentially the same program, but with a different index.

In his famous 1936 paper, Turing showed there is a partial computable function $U \colon \mathbb{N}^2 \to \mathbb{N}$ that encodes all other computable functions in the sense that, for every e, n,

$$U(e, n) = \Phi_e(n).$$

This function U is called a *universal partial computable function*, and it does essentially what computers do nowadays: You give them a(n index for a) program and an input, and they run it for you. We will not use U explicitly throughout the book, but we will constantly use the fact that we can computably list all programs and start running them one at the time, implicitly using U.

We identify subsets of $\mathbb{N}$ with their characteristic functions in $2^{\mathbb{N}}$, and we will move from one viewpoint to the other without even mentioning it. For instance, a set $A \subseteq \mathbb{N}$ is said to be *computable* if its characteristic function is.

An *enumeration* for set A is nothing more than an onto function $g \colon \mathbb{N} \to A$. A set A is *computably enumerable* (*c.e.*) if it has an enumeration that is computable. The empty set is computably enumerable too. Equivalently, a set is computably enumerable if it is the domain of a partial computable function. (If $A = range(g)$, then A is the domain of the partial function that, on input m, outputs the first n with $g(n) = m$ if it exists.) We denote

$$W_e = \{n \in \mathbb{N} : \Phi_e(n) \downarrow\} \quad \text{and} \quad W_{e,s} = \{n \in \mathbb{N} : \Phi_{e,s}(n) \downarrow\}.$$

As a convention, we assume that $W_{e,s}$ is finite, that is, only finitely many numbers can converge in less than s steps This makes sense because large numbers take more than s steps to be even read from the input tape. We sometimes use Lachlan's notation: $W_e[s]$ instead of $W_{e,s}$. In general, if a is an object built during a construction and whose value might change along the stages, we use $a[s]$ to denote its value at stage s.

Recall that a set is computable if and only if it and its complement are computably enumerable.

Sets and strings

The natural numbers are $\mathbb{N} = \{0, 1, 2, ...\}$. For $n \in \mathbb{N}$, we sometimes use n to denote the set $\{0, ..., n-1\}$. For instance, $2^{\mathbb{N}}$ is the set of functions from $\mathbb{N}$ to $\{0, 1\}$, which we will sometimes refer to as infinite binary sequences or infinite binary strings. For any set X, we use $X^{<\mathbb{N}}$ to denote the set of finite tuples of elements from X, which we call *strings* when $X = 2$ or $X = \mathbb{N}$. For $\sigma \in X^{<\mathbb{N}}$ and $\tau \in X^{\leq\mathbb{N}}$, we use $\sigma^\frown\tau$ to denote the concatenation of these sequences. We use $\sigma \subseteq \tau$ to denote that σ is a an initial segment of τ, that is, that $|\sigma| \leq |\tau|$ and $\sigma(n) = \tau(n)$ for all $n < |\sigma|$. Given $f \in X^{\leq\mathbb{N}}$ and $n \in \mathbb{N}$, we use $f \upharpoonright n$ to denote the initial segment of f of length n. We use $f \Vdash n$ for the initial segment of length $n + 1$. For a set $A \subseteq \mathbb{N}$, the complement of A is denoted by $\bar{A}$.

Given $f, g \in X^{\mathbb{N}}$, we use $f \oplus g$ for the function $(f \oplus g)(2n) = f(n)$ and $(f \oplus g)(2n+1) = g(n)$. We can extend this to ω sums and define $\bigoplus_n f_n$ to be the function defined by $(\bigoplus_n f_n)(m, k) = f_m(k)$. Conversely, we define $f^{[n]}$ to be the nth column of f, that is, $f^{[n]}(m) = f(n, m)$. All these definitions work for sets if we think in terms of their characteristic functions.

Reducibilities

There are various ways in which one can compare the complexity of sets of natural numbers. Depending on the context or application, some may be more appropriate than others.

Many-one reducibility. Given sets $A, B \subseteq \mathbb{N}$, we say that A is *many-one reducible* (or *m-reducible*) to B, and write $A \leq_m B$, if there is a computable function $f \colon \mathbb{N} \to \mathbb{N}$ such that $n \in A \iff f(n) \in B$ for all $n \in \mathbb{N}$. One should think of this reducibility as saying that all the information in A is directly encoded in B. Notice that the classes of computable sets and of c.e. sets are both closed downwards under $\leq_m$. A set B is said to be *c.e. complete* if it is c.e. and, for every other c.e. set A, $A \leq_m B$.

Two sets are *m-equivalent* if they are m-reducible to each other — denoted $A \equiv_m B$. This is an equivalence relation, and the equivalence classes are called *m-degrees*

There are, of course, various other ways to formalize the idea of one set encoding the information from another set. Many-one reducibility is somewhat restrictive in various ways: (1) to figure out if $n \in A$, one is allowed to only ask one question of the form "$m \in B$?"; (2) the answer to "$n \in A$?" has to be the same as the answer to "$f(n) \in B$?" Turing reducibility is much more flexible.

Turing reducibility. Given a function $f \colon \mathbb{N} \to \mathbb{N}$, we say that a partial function $g \colon \mathbb{N} \rightharpoonup \mathbb{N}$ is *partial f-computable* if it can be computed by a program that is allowed to use the function f as a primitive function during its computation; that is, the program can ask questions about the value of $f(n)$ for different n's and use the answers to make decisions while the program is running. The function f is called the *oracle* of this computation. For g and f total, we write $g \leq_T f$ when that is the case and say that g is *Turing reducible* to f. The class of partial f-computable functions can be

enumerated the same way as the class of the partial computable functions. Programs that are allowed to query an oracle are called *Turing operators*. We list them as Φ_0, Φ_1,... and we write $\Phi_e^f(n)$ for the output of the eth Turing operator on input n when it uses f as oracle. Notice that Φ_e represents a fixed program that can be used with different oracles. We can view the programs Φ_e we defined before as $\Phi_e^\emptyset$, where $\emptyset$ is the constant function equal zero, the characteristic function of the empty set.

For a fixed input n, if $\Phi_e^f(n)$ converges, it does so after a finite number of steps s. As a convention, let us assume that in just s steps, it is only possible to read at most the first s entries from the oracle. Thus, if σ is a finite substring of f of length greater than s, we could calculate $\Phi_e^\sigma(n)$ without ever noticing that the oracle is not an infinite string.

> Convention: For $\sigma \in \mathbb{N}^{<\mathbb{N}}$, $\Phi_e^\sigma(n)$ is shorthand for $\Phi_{e,|\sigma|}^\sigma(n)$, which runs for at most $|\sigma|$ stages.

Notice that given e, σ, n, it is computable to decide if $\Phi_e^\sigma(n) \downarrow$.

As the class of partial computable functions, the class of partial X-computable functions contains the basic functions; is closed under composition, recursion, and minimization; and can be listed in such a way that we have a universal partial X-computable function. In practice, with very few exceptions, those are the only properties we use of computable functions. This is why almost everything we can prove about computable functions, we can prove about X-computable functions too. This translation is called *relativization*. All notions whose definition are based on the notion of partial computable function can be relativized by using the notion of partial X-computable function instead. For instance, the notion of c.e. set can be relativized to that of X-c.e. set: These are the sets which are the images of X-computable functions (or empty), or equivalently, the domains of partial X-computable functions. We use W_e^X to denote the domain of Φ_e^X.

When two functions are Turing reducible to each other, we say that they are *Turing equivalent*, which we denote by $\equiv_T$. This is an equivalence relation, and the equivalence classes are called *Turing degrees*.

Computable operators can be encoded by computable subsets of $\mathbb{N}^{<\mathbb{N}} \times \mathbb{N} \times \mathbb{N}$. Given $\Phi \subseteq \mathbb{N}^{<\mathbb{N}} \times \mathbb{N} \times \mathbb{N}$, $f \in \mathbb{N}^{\mathbb{N}}$, n, m, we say that

$$\Phi^f(n) = m \iff (\exists \sigma \subset f)\ (\sigma, n, m) \in \Phi.$$

We then have that g is computable in f if and only if there is a computable subset $\Phi \subseteq \mathbb{N}^{<\mathbb{N}} \times \mathbb{N} \times \mathbb{N}$ such that $\Phi^f(n) = g(n)$ for all $n \in \mathbb{N}$. One can show that every c.e. operator $\Phi \subseteq \mathbb{N}^{<\mathbb{N}} \times \mathbb{N} \times \mathbb{N}$ is equivalent to a computable one.

Enumeration reducibility. Recall that an enumeration of a set A is just an onto function $f \colon \mathbb{N} \to A$. Given $A, B \subseteq \mathbb{N}$, we say that A is *enumeration reducible* (or *e-reducible*) to B, and write $A \leq_e B$, if every enumeration of B computes an enumeration of A. Selman [**Sel71**] showed that we can make this reduction uniformly: $A \leq_e B$ if and only if there is a Turing operator Φ such that, for every enumeration f of B, Φ^f is an enumeration of B. Another way of defining enumeration reducibility is via *enumeration operators*: An enumeration operator is a c.e. set of pairs Θ that acts as follows: For $B \subseteq \mathbb{N}$,

$$\Theta^B = \{n : (\exists D \subseteq_{fin} B)\ (\ulcorner D \urcorner, n) \in \Theta\}.$$

Selman also showed that $A \leq_e B$ if and only if there is an enumeration operator Θ such that $A = \Theta^B$.

The Turing degrees embed into the enumeration degrees via the map $\iota(A) = A \oplus \bar{A}$. It is not hard to show that $A \leq_T B \iff \iota(A) \leq_e \iota(B)$.

Positive-tt reducibility. We say that A *positively-tt reduces* to B, and write $A \leq_{ptt} B$, if there is a computable function $f\colon \mathbb{N} \to (\mathbb{N}^{<\mathbb{N}})^{<\mathbb{N}}$ such that, for every $n \in \mathbb{N}$, $n \in A$ if and only if there is an $i < |f(n)|$ such that every entry of $f(n)(i)$ is in B. That is,

$$n \in A \iff \bigvee_{i<|f(n)|} \bigwedge_{j<|f(n)(i)|} f(n)(i)(j) \in B.$$

Notice that $\leq_{ptt}$ implies both Turing reducibility and enumeration reducibility, and is implied by many-one reducibility. In particular, the classes of computable sets and of c.e. sets are both closed downwards under $\leq_{ptt}$.

The Turing jump. Let K be the domain of the universal partial computable functions. That is,

$$K = \{(e, n) : \Phi_e(n)\downarrow\} = \bigoplus_e W_e.$$

K is called the *halting problem*. It is not hard to see that K is c.e. complete. Using a standard diagonalization argument, one can show that K is not computable. (If it were computable, so would be the set $A = \{e : (e, e) \notin K\}$. But then $A = W_e$ for some e, and we would have that $e \in A \iff (e, e) \notin K \iff e \notin W_e \iff e \notin A$.) It is common to define K as $\{e : \Phi_e(e)\downarrow\}$ instead; it is not hard to see that the two definitions give m-equivalent sets. We will use whichever is more convenient in each situation.

We can relativize this definition and, given a set X, define the *Turing jump* of X as

$$X' = \{e \in \mathbb{N} : \Phi_e^X(e)\downarrow\}.$$

Relativizing the properties of K, we get that X' is X-c.e.-complete, that $X \leq_T X'$, and that $X' \not\leq_T X$. The Turing degree of X' is strictly above that of X — this is why it is called a jump. The jump defines an operation on the Turing degrees. Furthermore, for $X, Y \subseteq \omega$, $X \equiv_T Y \iff f(X) \equiv_m f(Y)$.

Vocabularies and languages

Let us quickly review the basics about vocabularies and structures. Our vocabularies will always be countable. Furthermore, except for a few occasions, they will always be computable.

A *vocabulary* τ consists of three sets of symbols $\{\mathsf{R}_i : i \in I_R\}$, $\{\mathsf{f}_i : i \in I_F\}$, and $\{\mathsf{c}_i : i \in I_C\}$; and two functions $a_R\colon I_R \to \mathbb{N}$ and $a_F\colon I_F \to \mathbb{N}$. Each of I_R, I_F, and I_C is an initial segment of $\mathbb{N}$. The symbols R_i, f_i, and c_i represent *relations*, *functions,* and *constants*, respectively. For $i \in I_R$, $a_R(i)$ is the arity of R_i, and for $i \in I_F$, $a_F(i)$ is the arity of f_i.

A vocabulary τ is *computable* if the arity functions a_R and a_F are computable. This only matters when τ is infinite; finite vocabularies are trivially computable.

Given such a vocabulary τ, a τ*-structure* is a tuple

$$\mathcal{M} = (M; \{R_i^{\mathcal{M}} : i \in I_R\}, \{f_i^{\mathcal{M}} : i \in I_F\}, \{c_i^{\mathcal{M}} : i \in I_C\}),$$

where M is just a set, called the *domain* of $\mathcal{M}$, and the rest are interpretations of the symbols in τ. That is, $R_i^{\mathcal{M}} \subset M^{a_R(i)}$, $f_i^{\mathcal{M}} \colon M^{a_F(i)} \to M$, and $c_i^{\mathcal{M}} \in M$. A *structure* is a τ-structure for some τ.

Given a vocabulary τ, we define various languages over it. First, recursively, a τ-*term* is either a variable x, a constant symbol c_i, or a function symbol applied to other terms, that is, $\mathsf{f}_i(t_1, ..., t_{a_F(i)})$, where each t_i is a τ-term we have already built. The *atomic* τ-*formulas* are the ones of the form $\mathsf{R}_i(t_1, ..., t_{a_R(i)})$ or $t_1 = t_2$, where each t_i is a τ-term. A τ-*literal* is either a τ-atomic formula or a negation of a τ-atomic formula. A τ-*quantifier-free formula* is build out of literals using conjunctions, disjunctions, and implications. If we also close under existential quantification, we get the τ-*existential formulas*. Every τ-existential formula is equivalent to one of the form $\exists x_1 \cdots \exists x_k \, \varphi$, where φ is quantifier-free. A τ-*universal formula* is one equivalent to one of the form $\forall x_1 \cdots \forall x_k \, \varphi$, where φ is quantifier-free. A τ-*elementary formula* is built out of quantifier-free formulas using existential and universal quantifiers.

The arithmetic hierachy

Consider the structure $(\mathbb{N}; 0, 1, +, \times, \leq)$. In this vocabulary, the *bounded formulas* are build out of the quantifier-free formulas using bounded quantifiers of the form $\forall x < y$ and $\exists x < y$. A Σ_1^0 formula is one of the form $\exists x \, \varphi$, where φ is bounded; and a Π_1^0 formula is one of the form $\forall x \, \varphi$, where φ is bounded. Coding tuples by single natural numbers, one can show that formulas of the form $\exists x_0 \exists x_1 \cdots ... \exists x_k \, \varphi$ are equivalent to Σ_1^0 formulas. It requires some work, but one can show that a set $A \subseteq \mathbb{N}$ is c.e. if and only if it can be defined by a Σ_1^0 formula. Thus, a set is computable if it is Δ_1^0, that is, if it can be defined by both a Σ_1^0 and Π_1^0 formulas.

By recursion, we define the Σ_{n+1}^0 formulas as those of the form $\exists x \, \varphi$, where φ is Π_n^0; and the Π_{n+1}^0 formulas as those of the form $\forall x \, \varphi$, where φ is Σ_n^0. A set is Δ_n^0 if it can be defined by both a Σ_n^0 formula and a Π_n^0 formula. Again, in the definition of Σ_{n+1}^0 formulas, using one existential quantifier or many makes no difference. What matters is the number of alternations of quantifiers.

Part 1

Introductory Topics

CHAPTER I

Structures

Algorithms, Turing machines, and modern computer programs all work with finitary objects, objects that usually can be encoded by finite binary strings or just by natural numbers. For this reason, computability theory concentrates on the study the of complexity of sets of natural numbers. When we want to study a countable mathematical structure, the first approach is to set the domain of the structure to be a subset of the natural numbers and then borrow the tools we already have from computability theory. There is one issue that comes up here: There might be many bijections between the domain of a structure and the natural numbers, inducing many different *presentations* of the structure with different computability-theoretic properties. The interplay between properties of presentations (computational properties) and properties of isomorphism types (structural properties) is one of the main themes of computable structure theory.

We start this chapter by introducing various ways of presenting structures in ways we can analyze their computational complexity. These different types of presentations are essentially equivalent, and the distinctions are purely technical and not deep. However, we need to introduce them as basic background as they will allow us to be precise later. At the end of the chapter we prove Knight's theorem that non-trivial structures have presentations that code any given set.

I.1. Presentations

All the structures we consider are countable. So, unless otherwise stated, "structure" means "countable structure." Furthermore, we usually assume the domain of our structures to be a subset of $\mathbb{N}$. This will allow us to use everything we know about computable functions on $\mathbb{N}$ to study structures.

Definition I.1.1. An *ω-presentation* is nothing more than a structure whose domain is $\mathbb{N}$. Given a structure $\mathcal{A}$, when we refer to *an ω-presentation of $\mathcal{A}$* or to a *copy* of $\mathcal{A}$, we mean an ω-presentation $\mathcal{M}$ which is isomorphic to $\mathcal{A}$. An ω-presentation $\mathcal{M}$ is *computable* if all its relations, functions, and constants are uniformly computable; that is, if the set $\tau^{\mathcal{M}}$, defined as

$$\tau^{\mathcal{M}} = \bigoplus_{i\in I_R} R_i^{\mathcal{M}} \oplus \bigoplus_{i\in I_F} F_i^{\mathcal{M}} \oplus \bigoplus_{i\in I_C} \{c_i^{\mathcal{M}}\}, \tag{1}$$

is computable. Note that via standard coding, we can think of $\tau^{\mathcal{M}}$ as a subset of $\mathbb{N}$.

I.1.1. Atomic diagrams. Another standard way of defining the computability of a presentation is via its atomic diagram. Let $\{\varphi_i^{\text{at}} : i \in \mathbb{N}\}$ be an effective enumeration of all atomic τ-formulas on the variables $\{x_0, x_1, ...\}$. (An *atomic τ-formula* is one of the form $\mathsf{R}(\mathsf{t}_1, ..., \mathsf{t}_a)$, where R is either "=" or R_j for $j \in I_R$, and each t_i is a term built out of the function, constant, and variable symbols.)

DEFINITION I.1.2. The *atomic diagram* of an ω-presentation $\mathcal{M}$ is the infinite binary string $D(\mathcal{M}) \in 2^{\mathbb{N}}$ defined by

$$D(\mathcal{M})(i) = \begin{cases} 1 & \text{if } \mathcal{M} \models \varphi_i^{\text{at}}[x_j \mapsto j : j \in \mathbb{N}] \\ 0 & \text{otherwise.} \end{cases}$$

It is not hard to see that $D(\mathcal{M})$ and $\tau^{\mathcal{M}}$ are Turing equivalent. We will often treat the ω-presentation $\mathcal{M}$, the real $\tau^{\mathcal{M}}$, and the real $D(\mathcal{M})$ as the same thing. For instance, we define the *Turing degree of the ω-presentation $\mathcal{M}$* to be the Turing degree of $D(\mathcal{M})$. When we say that $\mathcal{M}$ *is computable from a set X*, that *a set X is computable from $\mathcal{M}$*, that $\mathcal{M}$ is Δ^0_2, that $\mathcal{M}$ is *arithmetic*, that $\mathcal{M}$ is *low*, etc., we mean $D(\mathcal{M})$ instead of $\mathcal{M}$.

Let us also point out that the quantifier-free diagram, defined like the atomic diagram but with quantifier-free formulas, is also Turing equivalent to $D(\mathcal{M})$. We let the reader verify this fact.

I.1.2. An example. Unless it is trivial, a structure will have many different ω-presentations — continuum many actually (see Theorem I.2.1) — and these different ω-presentations will have different computability theoretic properties. For starters, some of them may be computable while others may not. But even among the computable copies of a single structure one may find different computability behaviors.

Consider the linear ordering $\mathcal{A} = (\mathbb{N}; \leq)$, where $\leq$ is the standard ordering on the natural numbers. We can build another ω-presentation $\mathcal{M} = (\mathbb{N}; \leq_M)$ of $\mathcal{A}$ as follows. (Recall that $\{k_i : i \in \mathbb{N}\}$ is a computable one-to-one enumeration of $0'$). First, order the even natural numbers in the natural way: $2n \leq_M 2m$ if $n \leq m$. Second, place the odd number $2s+1$ right in between $2k_s$ and $2k_s+2$, that is, let $2k_s \leq_M 2s+1 \leq_M 2k_s+2$. Using transitivity we can then define $\leq_M$ on all of pairs of numbers. Thus $2n <_M 2s+1$ if and only if $n < k_s$, and $2s+1 <_M 2t+1$ if and only if $k_s < k_t$.

We then have that $\mathcal{A}$ and $\mathcal{M}$ are two computable ω-presentations of the same structure. However, computationally, they behave quite differently. For instance, the successor function is computable in $\mathcal{A}$ but not in $\mathcal{M}$: In $\mathcal{A}$, $Succ^{\mathcal{A}}(n) = n+1$ is clearly computable. On the other hand, in $\mathcal{M}$, $Succ^{\mathcal{M}}(2n) = 2n+2$ if and only if there is no odd number placed $\leq_M$-in-between $2n$ and $2n+2$, which occurs if and only if $n \notin 0'$. Therefore, $Succ^{\mathcal{M}}$ computes $0'$.

The reason $\mathcal{A}$ and $\mathcal{M}$ can behave differently despite being isomorphic is that they are not *computably isomorphic*: There is no computable isomorphism between them because, if there were, we could use $Succ^{\mathcal{A}}$ and the isomorphism to compute $Succ^{\mathcal{M}}$, contradicting that $Succ^{\mathcal{M}}$ computes $0'$.

I.1.3. Relaxing the domain. In many cases, it will be useful to consider structures whose domain is a subset of $\mathbb{N}$. We call those ($\subseteq \omega$)*-presentations*. If M, the domain of $\mathcal{M}$, is a proper subset of $\mathbb{N}$, we can still define $D(\mathcal{M})$ by letting $D(\mathcal{M})(i) = 0$ if φ_i^{at} mentions a variable x_j with $j \notin M$. In this case, we have

$$D(\mathcal{M}) \equiv_T M \oplus \tau^{\mathcal{M}}.$$

To see that $D(\mathcal{M})$ computes M, notice that, for $j \in \mathbb{N}$, $j \in M \leftrightarrow D(\mathcal{M})(\ulcorner x_j = x_j \urcorner) = 1$, where $\ulcorner \varphi \urcorner$ is the index of the atomic formula φ in the enumeration $\{\varphi_i^{\text{at}} : i \in \mathbb{N}\}$.

The following observation will simplify many of our constructions later on.

OBSERVATION I.1.3. We can always associate to an infinite ($\subseteq \omega$)-presentation $\mathcal{M}$, an isomorphic ω-presentation $\mathcal{A}$: If $M = \{m_0 < m_1 < m_2 < \cdots\} \subseteq \mathbb{N}$, we can use the bijection $i \mapsto m_i \colon \mathbb{N} \to M$ to get a copy $\mathcal{A}$ of $\mathcal{M}$, now with domain $\mathbb{N}$. Since this bijection is computable in M, it is not hard to see that $D(\mathcal{A}) \leq_T D(\mathcal{M})$, and furthermore that $D(\mathcal{A}) \oplus M \equiv_T D(\mathcal{M})$.

One of the advantages of ($\subseteq \omega$)-presentations is that they allow us to present finite structures.

I.1.4. Relational vocabularies. A vocabulary is *relational* if it has no function or constant symbols, and has only relational symbols. Every vocabulary τ can be made into a relational one, $\tilde{\tau}$, by replacing each n-ary function symbol by an $(n+1)$-ary relation symbol coding the graph of the function, and each constant symbol by a 1-ary relation symbol coding it as a singleton. Depending on the situation, this change in vocabulary might be more or less relevant. For instance, the class of quantifier-free definable sets changes, but the class of $\exists$-definable sets does not (see Exercise I.1.4). For most computational properties, this change is nonessential; for instance, if $\mathcal{M}$ is an ω-presentation, and $\tilde{\mathcal{M}}$ is the associated ω-presentation of $\mathcal{M}$ as a $\tilde{\tau}$-structure, then $D(\mathcal{M}) \equiv_T D(\tilde{\mathcal{M}})$ (as it follows from Exercise I.1.4). Because of this, and for the sake of simplicity, we will often restrict ourselves to relational languages.

EXERCISE I.1.4. Show that the $\exists$-*diagram of* $\mathcal{M}$ as a τ structure is 1-equivalent to its $\exists$-diagram as a $\tilde{\tau}$ diagram. More concretely, let $\{\varphi_i^{\exists} : i \in \mathbb{N}\}$ and $\{\tilde{\varphi}_i^{\exists} : i \in \mathbb{N}\}$ be the standard effective enumerations of the existential τ-formulas and the existential $\tilde{\tau}$-formulas on the variables $x_0, x_1, \ldots$. Show that

$$\{i \in \mathbb{N} : \mathcal{M} \models \varphi_i^{\exists}[x_j \mapsto j : j \in \mathbb{N}]\} \quad \equiv_1 \quad \{i \in \mathbb{N} : \mathcal{M} \models \tilde{\varphi}_i^{\exists}[x_j \mapsto j : j \in \mathbb{N}]\}.$$

I.1.5. Finite Structures and Approximations. We can represent finite structures using ($\subseteq \omega$)-presentations. However, when working with infinitely many finite structures at once, we often want to be able to compute things about them uniformly, for instance the sizes of the structures — something we could not do from ($\subseteq \omega$)-presentations. For that reason, we sometimes consider ($\sqsubseteq \omega$)-presentations, which are ($\subseteq \omega$)-presentations whose domains are initial segments of $\mathbb{N}$. Given a finite ($\sqsubseteq \omega$)-presentation, we can easily find the first k that is not in the domain of the structure.

When τ is a finite vocabulary, finite τ-structures can be coded by a finite amount of information. Suppose $\mathcal{M}$ is a finite τ-structure with domain $\{0, \ldots, k-1\}$, and τ is a finite relational vocabulary. Then there are only finitely many atomic τ-formulas on the variables $x_0, \ldots, x_{k-1}$, let us say ℓ_k of them. Assume the enumeration $\{\varphi_i^{\text{at}} : i \in \mathbb{N}\}$ of the atomic τ-formulas is such that those ℓ_k formulas come first, and the formulas mentioning other variables come later. Then $D(\mathcal{M})$ is determined by the finite binary string of length ℓ_k that codes the values of those formulas. We will often assume $D(\mathcal{M})$ *is* that string.

When dealing with infinite structures, very often we will want to approximate them using finite substructures. We need to take care of two technical details. First, if τ is an infinite vocabulary, we need to approximate it using finite sub-vocabularies. We assume that all computable vocabularies τ come with an associated effective approximation

$\tau_0 \subseteq \tau_1 \subseteq \cdots \subseteq \tau$, where each τ_s is finite and $\tau = \bigcup_s \tau_s$. In general and unless otherwise stated, we let τ_s consist of the first s relation, constant and function symbols in τ, but in some particular cases, we might prefer other approximations. For instance, if τ is already finite, we usually prefer to let $\tau_s = \tau$ for all s. Second, to be able to approximate a τ-structure $\mathcal{M}$ using τ_s-substructures, we need the τ_s-reduct of $\mathcal{M}$ to be *locally finite*, i.e., every finite subset generates a finite substructure. To avoid unnecessary complications, we will just assume τ is relational and, in particular, locally finite. Even if τ is not originally relational, we can make it relational as in Section I.1.4.

Definition I.1.5. Given an ω-presentation $\mathcal{M}$, we let $\mathcal{M}_s$ be the finite τ_s-substructure of $\mathcal{M}$ with domain $\{0, ..., s-1\}$. We call the sequence $\{\mathcal{M}_s : s \in \mathbb{N}\}$ a *finite approximation* of $\mathcal{M}$. We identify this sequence with the sequence of codes $\{D(\mathcal{M}_s) : s \in \mathbb{N}\} \subseteq 2^{<\mathbb{N}}$, which allows us to consider its computational complexity.

In general, when we refer to a $\tau_{|\cdot|}$-*structure*, we mean a τ_s-structure where s is the size of the structure itself. For instance, the structures $\mathcal{M}_s$ above are all $\tau_{|\cdot|}$-structures.

Observation I.1.6. For each s, $D(\mathcal{M}_s) = D(\mathcal{M}) \upharpoonright \ell_s$, and hence $D(\mathcal{M}) = \bigcup_s D(\mathcal{M}_s)$.

Thus, having an ω-presentation is equivalent to having a finite approximation of a structure $\mathcal{M}$. This is why, when we are working with an ω-presentation, we often visualize the structure as being given to us little by little.

As a useful technical device, we define the atomic diagram of a finite tuple as the set of atomic formulas true about the tuple restricted to the smaller vocabulary. Again, we assume that τ is relational and that the enumeration of the τ-atomic formulas used in the definition of $D(\mathcal{M})$ has the following property: For each s, the τ_s-atomic formulas on the variables $\{x_0, ..., x_{s-1}\}$ are listed before the rest; that is, they are $\varphi_0^{\text{at}}, ..., \varphi_{\ell_s-1}^{\text{at}}$ for some $\ell_s \in \mathbb{N}$.

Definition I.1.7. Let $\mathcal{M}$ be a τ-structure and let $\bar{a} = \langle a_0, ..., a_{s-1}\rangle \in M^s$. We define the *atomic diagram of* $\bar{a}$ *in* $\mathcal{M}$*, denoted* $D_{\mathcal{M}}(\bar{a})$, as the string in 2^{ℓ_s} such that

$$D_{\mathcal{M}}(\bar{a})(i) = \begin{cases} 1 & \text{if } \mathcal{M} \models \varphi_i^{\text{at}}[x_j \mapsto a_j, j < s], \\ 0 & \text{otherwise.} \end{cases}$$

So, if $\mathcal{M}$ were an ω-presentation and $a_0, ...a_s, ...$ were the elements $0, ..., s, ... \in M = \mathbb{N}$, then $D_{\mathcal{M}}((a_0, ..., a_{s-1})) = D(\mathcal{M}_s)$ as in Definition I.1.5.

Observation I.1.8. For every $s \in \mathbb{N}$ and every $\sigma \in 2^{\ell_s}$, there is a quantifier-free τ-formula $\varphi_\sigma^{\text{at}}(x_0, ..., x_{s-1})$ such that, for every structure $\mathcal{A}$ and tuple $\bar{a} \in A^s$,

$$\sigma = D_{\mathcal{A}}(\bar{a}) \iff \mathcal{A} \models \varphi_\sigma^{\text{at}}(\bar{a}),$$

Furthermore, for every $\sigma \in 2^{<\mathbb{N}}$ with $\ell_s \geq |\sigma|$, there is a quantifier-free formula $\varphi_\sigma^{\text{at}}(x_0, ..., x_{s-1})$ such that $\sigma \subseteq D_{\mathcal{A}}(\bar{a}) \iff \mathcal{A} \models \varphi_\sigma^{\text{at}}(\bar{a})$ for every $\mathcal{A}$ and $\bar{a} \in A^s$, namely

$$\varphi_\sigma^{at}(\bar{x}) \quad \equiv \quad \left(\bigwedge_{i<|\sigma|, \sigma(i)=1} \varphi_i^{at}(\bar{x}) \right) \wedge \left(\bigwedge_{i<|\sigma|, \sigma(i)=0} \neg\varphi_i^{at}(\bar{x}) \right).$$

I.1.6. Congruence structures. It will often be useful to consider structures where equality is interpreted by an equivalence relation. A *congruence τ-structure* is a structure $\mathcal{M} = (M; =^{\mathcal{M}}, \{R_i^{\mathcal{M}} : i \in I_R\}, \{f_i^{\mathcal{M}} : i \in I_F\}, \{c_i^{\mathcal{M}} : i \in I_C\})$, where $=^{\mathcal{M}}$ is an equivalence relation on M, and the interpretations of all the τ-symbols are invariant under $=^{\mathcal{M}}$ (that is, if $\bar{a} =^{\mathcal{M}} \bar{b}$, then $\bar{a} \in R_i^{\mathcal{M}} \iff \bar{b} \in R_i^{\mathcal{M}}$ and $f_j^{M}(\bar{a}) =^{\mathcal{M}} f_j(\bar{b})$ for all $i \in I_R$ and $j \in I_F$). If $M = \mathbb{N}$, we say that $\mathcal{M}$ is a *congruence ω-presentation*. We can then define $D(\mathcal{M})$ exactly as in Definition I.1.2, using $=^{\mathcal{M}}$ to interpret equality.

Given a congruence τ-structure, one can always take the quotient $\mathcal{M}/=^{\mathcal{M}}$ and get a τ-structure where equality is the standard $\mathbb{N}$-equality. To highlight the difference, we will sometimes use the term *injective ω-presentations* when equality is $\mathbb{N}$-equality.

Lemma I.1.9. *Given a congruence ω-presentation $\mathcal{M}$ with infinitely many equivalence classes, the quotient $\mathcal{M}/=^{\mathcal{M}}$ has an injective ω-presentation $\mathcal{A}$ computable from $D(\mathcal{M})$. Furthermore, the natural projection $\mathcal{M} \to \mathcal{A}$ is also computable from $D(\mathcal{M})$.*

Proof. All we need to do is pick a representative for each $=^{\mathcal{M}}$-equivalence class in a $D(\mathcal{M})$-computable way. Just take the $\mathbb{N}$-least element of each class: Let

$$A = \{a \in M : (\forall b \in M, b <_{\mathbb{N}} a)\ b \neq^{M} a\}$$

be the domain of $\mathcal{A}$. Define the functions and relations in the obvious way to get a $(\subseteq \omega)$-presentation of $\mathcal{M}$. To get an ω-presentation, use Observation I.1.3. □

Therefore, from a computational viewpoint, there is no real difference in considering congruence structures or injective structures.

I.1.7. Enumerations. Assume τ is a relational vocabulary. An *enumeration of a structure $\mathcal{M}$* is just an onto map $g\colon \mathbb{N} \to M$. To each such enumeration we can associate a congruence ω-presentation $g^{-1}(\mathcal{M})$ by taking the *pull-back* of $\mathcal{M}$ through g:

$$g^{-1}(\mathcal{M}) = (\mathbb{N}; \sim, \{R_i^{g^{-1}(\mathcal{M})} : i \in I_R\}),$$

where $a \sim b \iff g(a) = g(b)$ and $R_i^{g^{-1}(\mathcal{M})} = g^{-1}(R_i^{\mathcal{M}}) \subseteq \mathbb{N}^{a(i)}$. The assumption that τ is relational was used here so that the pull-backs of functions and constants are not multi-valued. Let us remark that if g is injective, then $\sim$ becomes $=_{\mathbb{N}}$, and hence $g^{-1}(\mathcal{M})$ is an injective ω-presentation. In this case, the assumption that τ is relational is not important, as we can always pull-back functions and constants through bijections.

It is not hard to see that

$$D(g^{-1}(\mathcal{M})) \leq_T g \oplus D(\mathcal{M}).$$

Furthermore, $D(g^{-1}(\mathcal{M})) \leq_T g \oplus \tau^{\mathcal{M}}$, where $\tau^{\mathcal{M}}$ is as in Definition I.1.1. As a corollary we get the following lemma.

Lemma I.1.10. *Let $\mathcal{A}$ be a computable structure on a relational language and M be an infinite c.e. subsets of A. Then, the substructure $\mathcal{M}$ of $\mathcal{A}$ with domain M has a computable ω-presentation.*

Proof. Just let g be a computable injective enumeration of M and consider $g^{-1}(\mathcal{M})$. □

As a corollary of the lemma, we get that, if a structure is defined as a substructure of a computable ω-presentation generated by a a c.e. subset, then the new structure has a computable copy.

Throughout the book, there will be many constructions where we need to build a copy of a given structure with certain properties. In most cases, we will do it by building an enumeration of the structure and then taking the pull-back. The following observation will allow us to approximate the atomic diagram of the pull-back.

OBSERVATION I.1.11. Let g be an enumeration of $\mathcal{M}$. Notice that for every tuple $\bar{a}$, $D_{g^{-1}(\mathcal{M})}(\bar{a}) = D_{\mathcal{M}}(g(\bar{a}))$. For each k, use $g \upharpoonright k$ to denote the tuple $\langle g(0), ..., g(k-1)\rangle \in M^k$. Then $D_{g^{-1}(\mathcal{M})}(\langle 0, ..., k-1\rangle) = D_{\mathcal{M}}(g \upharpoonright k)$ and the diagram of the pull-back can be calculated in terms of the diagrams of tuples in $\mathcal{M}$ as follows:

$$D(g^{-1}(\mathcal{M})) = \bigcup_{k\in\mathbb{N}} D_{\mathcal{M}}(g \upharpoonright k).$$

I.2. Presentations that code sets

In this section, we show that the Turing degrees of ω-presentations of a non-trivial structure can be arbitrarily high. Furthermore, we prove a well-known theorem of Julia Knight's that states that the set of Turing degrees of the ω-presentations of a structure is upwards closed. This set of Turing degrees is called the *degree spectrum* of the structure, and we will study it in detail in Chapter V. Knight's theorem is only true if the structure is non-trivial: A structure $\mathcal{A}$ is *trivial* if there is a finite tuple such that every permutation of the domain fixing that tuple is an automorphism. Notice that these structures are essentially finite in the sense that anything relevant about them is coded in that finite tuple.

THEOREM I.2.1 (Knight [**Kni98**]). *Suppose that X can compute an ω-presentation of a τ-structure $\mathcal{M}$ which is non-trivial. Then there is an ω-presentation $\mathcal{A}$ of $\mathcal{M}$ of Turing degree X.*

Before proving the theorem, let us remark that if, instead of an ω-presentation, we wanted a ($\subseteq \omega$)-presentation or a congruence ω-presentation, it would be very easy to code X into either the domain or the equality relation of $\mathcal{A}$: Recall that $\mathcal{D}(\mathcal{A}) = A \oplus (=^{\mathcal{A}}) \oplus \tau^{\mathcal{A}}$. Requiring $\mathcal{A}$ to be an injective ω-presentation forces us to code X in the structural part of $\mathcal{A}$, namely $\tau^{\mathcal{A}}$.

PROOF. We will build an X-computable injective enumeration g of $\mathcal{M}$ and let $\mathcal{A} = g^{-1}(\mathcal{M})$. That is already enough to give us $D(\mathcal{A}) \leq_T X$; the actual work comes from ensuring that $D(\mathcal{A}) \geq_T X$. We build g as a limit $\bigcup_s \bar{p}_s$, where the $\bar{p}_s$ form a nested sequence of tuples $\bar{p}_0 \subseteq \bar{p}_1 \subseteq \cdots$ in $M^{<\mathbb{N}}$, getting $\bigcup_s \bar{p}_s \in M^{\mathbb{N}}$. Recall from Observation I.1.11 that we can approximate the atomic diagram of $\mathcal{A}$ by the atomic diagrams of the tuples $\bar{p}_s$:

$$D(\mathcal{A}) = \bigcup_{s\in\mathbb{N}} D_{\mathcal{M}}(\bar{p}_s).$$

Let $\bar{p}_0 = \emptyset$. Suppose now we have already defined already $\bar{p}_s$. At stage $s+1$, we build $\bar{p}_{s+1} \supseteq \bar{p}_s$ with the objective of coding $X(s) \in \{0, 1\}$ into $D(\mathcal{A})$. The idea for coding $X(s)$ is as follows: We would like to find $a, b \in M \setminus \bar{p}_s$ such that $D_{\mathcal{M}}(\bar{p}_s a) \neq D_{\mathcal{M}}(\bar{p}_s b)$.

Suppose we find them and $D_{\mathcal{M}}(\bar{p}_s a) <_{lex} D_{\mathcal{M}}(\bar{p}_s b)$, where $\leq_{lex}$ is the lexicographical ordering on strings in $2^{<\mathbb{N}}$. Then, depending on whether $X(s) = 0$ or 1, we can define $\bar{p}_{s+1}$ to be either $\bar{p}_s ab$ or $\bar{p}_s ba$. To decode $X(s)$, all we have to do is compare the binary strings $D_{g^{-1}(\mathcal{M})}(0, ..., k-1,\ k)$ and $D_{g^{-1}(\mathcal{M})}(0, ..., k-1,\ k+1)$, where $k = |\bar{p}_s|$.

The problem with this idea is that there may not be such a and b, and $D_{\mathcal{M}}(\bar{p}_s a)$ might be the same for all $a \in M$. Since $\mathcal{M}$ is non-trivial, we know there is some bijection of M preserving $\bar{p}_s$ which is not an isomorphism, and hence there exist tuples $\bar{a}$ and $\bar{b} \in (M \setminus \bar{p}_s)^{<\mathbb{N}}$ of the same length with $D_{\mathcal{M}}(\bar{p}_s \bar{a}) \neq D_{\mathcal{M}}(\bar{p}_s \bar{b})$. Furthermore, there are disjoint such $\bar{a}$ and $\bar{b}$ (if we take a tuple disjoint from $\bar{a}$ and $\bar{b}$, its diagram must be different from that of either $\bar{a}$ or $\bar{b}$ and we can replace it for $\bar{b}$ or $\bar{a}$ accordingly). So we search for such a pair of tuples $\bar{a}, \bar{b}$, say of length h. We also require the pair $\bar{a}, \bar{b}$ to be minimal, in the sense that $D_{\mathcal{M}}(\bar{p}_s a_0, ..., a_{i-1}) = D_{\mathcal{M}}(\bar{p}_s b_0, ..., b_{i-1})$ for $i - 1 \leq h - 1$. Suppose $D_{\mathcal{M}}(\bar{p}_s \bar{a}) <_{lex} D_{\mathcal{M}}(\bar{p}_s \bar{b})$ (otherwise replace $\bar{a}$ for $\bar{b}$ in what follows). If $X(s) = 0$, let $\tilde{p}_{s+1} = \bar{p}_s a_0 b_0 a_1 b_1, ..., a_{h-1} b_{h-1}$. If $X(s) = 1$, let $\tilde{p}_{s+1} = \bar{p}_s b_0 a_0 b_1 a_1, ..., b_{h-1} a_{h-1}$. Finally, to make sure g is onto, we let $\bar{p}_{s+1} = \tilde{p}_{s+1} c$, where c is the $\mathbb{N}$-least element of $M \setminus \tilde{p}_{s+1}$.

To recover X from $D(\mathcal{A})$, we need to also simultaneously recover the sequence of lengths $\{k_s : s \in \mathbb{N}\}$, where $k_s = |\bar{p}_s|$. Given k_s, we can compute k_{s+1} uniformly in $D(\mathcal{A})$ as follows: k_{s+1} is the least $k > k_s$ such that

$$D_{\mathcal{A}}(0, ..., k_s - 1,\ k_s, k_s + 2, k_s + 4, ..., k - 3) \neq$$
$$D_{\mathcal{A}}(0, ..., k_s - 1,\ k_s + 1, k_s + 3, k_s + 5, ..., k - 2).$$

Once we know which of these two binary strings is lexicographically smaller, we can tell if $X(s)$ is 0 or 1: It is 0 if the former one is $\leq_{lex}$-smaller than the latter one. □

Notice that for trivial structures, all presentations are isomorphic via computable bijections, and hence all presentations have the same Turing degree. When the vocabulary is finite, all trivial structures are computable.

CHAPTER II

Relations

A *relation* is just a set of tuples from a structure. The study of the complexity and definability of such a basic concept is one of the main components of computable structure theory. Many of the notions of computability on subsets of $\mathbb{N}$ can be extended to relations on a structure, but the space of relations is usually much richer than the space of subsets of $\mathbb{N}$, and understanding that space allows us to infer properties about the underlying structure. In this chapter we will introduce the analogs of the notions of c.e.ness, Turing reducibility, join and jump for the space of relations. These tools will then be used throughout the book.

From now on, unless otherwise stated, when we are given a structure, we are given an ω-presentation of a structure. Throughout this chapter, $\mathcal{A}$ always denotes an ω-presentation of a τ-structure.

II.1. Relatively intrinsic notions

We start by defining a notion of *c.e.-ness* for relations on a given structure. This will open the door for generalizing other notions of computability theory from subsets of $\mathbb{N}$ to relations on a structure.

II.1.1. R.i.c.e. relations. The idea we are trying to capture is what is behind the following examples:

Example II.1.1. Over a $\mathbb{Q}$-vector space $\mathcal{V}$, the relation LD $\subseteq V^{<\mathbb{N}}$ of linear dependence is always c.e. in $\mathcal{V}$ (more concretely, LD is the set of tuples $(v_0, ..., v_k) \in V^{<\mathbb{N}}$ of vectors which are linearly dependent). To enumerate LD in a $D(\mathcal{V})$-computable way, go through all the possible non-trivial $\mathbb{Q}$-linear combinations of $(v_0, ..., v_k)$, and if you find one that is equal to 0, enumerate $(v_0, ..., v_k)$ into LD.

Example II.1.2. Over a ring $\mathcal{R}$, the relation that holds of $(r_0, ..., r_k) \in R^{<\mathbb{N}}$ if the polynomial $r_0 + r_1x + ... + r_kx^k$ has a root is c.e. in $\mathcal{R}$: As in the previous example, search for a root of the polynomial going through all the possible values of $x \in R$, and if you ever find one that makes the polynomial 0, enumerate $(r_0, ..., r_k)$ into the relation.

Definition II.1.3. Let $\mathcal{A}$ be a structure. A relation $R \subseteq A^{<\mathbb{N}}$ is *relatively intrinsically computably enumerable* (*r.i.c.e.*) if, on every copy $(\mathcal{B}, R^{\mathcal{B}})$ of $(\mathcal{A}, R)$, we have that $R^{\mathcal{B}}$ (viewed as a subset of $\mathbb{N}^{<\mathbb{N}}$) is c.e. in $D(\mathcal{B})$.

The relations from Examples II.1.1 and II.1.2 are both r.i.c.e. A relation like linear independence, whose complement is r.i.c.e., is said to be *co-r.i.c.e.*

Notice that the notion of being r.i.c.e. is independent of the presentation of $\mathcal{A}$, and depends only on its isomorphism type.

Let us remark that we can view $(\mathcal{A}, R)$ as a structure, as defined in page xiii, by thinking of R as an infinite sequence of relations $\langle R_n : n \in \mathbb{N}\rangle$, where $R_n = R \cap A^n$ has arity n. The original definitions of r.i.c.e. (see [**AK00**, Page 165] [**Mon12**, Definition 3.1]) are only on n-ary relations for fixed n, but that is too restrictive for us. The reason we choose to define r.i.c.e. on subsets of $A^{<\mathbb{N}}$ is that it is the simplest setting that is fully general. This is the same reason we choose to develop computability theory on sets of natural numbers instead on the set of hereditarily finite sets: The natural numbers are simpler, and yet every finite object can be coded with a single natural number. We will get back to this point in Section II.4.

Example II.1.4. Let $\mathcal{A}$ be a linear ordering $(A; \leq)$. We say that x and $y \in A$ are *adjacent*, and write $\mathrm{Adj}(x, y)$, if $x < y$ and there is no element in between them. Notice that the complement of this relation, $\neg\mathrm{Adj}(a, b) \subseteq A^2$, is c.e. in $D(\mathcal{A})$: At stage s, we are monitoring the first s elements of the ω-presentation of $\mathcal{A}$, and if we see an element appear in between a and b, we enumerate the pair (a, b) into $\neg\mathrm{Adj}(a, b)$. This is also the case for any other ω-presentation of $\mathcal{A}$, which makes $\neg$Adj r.i.c.e. There is something intrinsic about $\neg$Adj that makes it c.e. in whatever ω-presentation we consider. In the case of $\neg$Adj, the reason is actually quite explicit: It has an $\exists$-definition, namely

$$\neg\mathrm{Adj}(x, y) \iff x \not< y \vee \exists z\ (x < z < y).$$

Example II.1.5. Consider a linear ordering with the adjacency relation as part of the structure $\mathcal{A} = (A; <, \mathrm{Adj})$. On it, consider the set R of pairs of elements from A for which the number of elements in between them is a number that belongs to $0'$. We note that $R \subseteq A^2$ is r.i.c.e.: Given $a, b \in A$, wait to find elements $a_1, ..., a_n$ with $\mathrm{Adj}(a, a_1) \wedge \mathrm{Adj}(a_1, a_2) \wedge ... \wedge \mathrm{Adj}(a_{n-1}, a_n) \wedge \mathrm{Adj}(a_n, b)$, and if we ever find them, wait to see if n enters $0'$, and if that ever happens, enumerate (a, b) into R. The relation R cannot be defined by an $\exists$-formula in the vocabulary $\{\leq, \mathrm{Adj}\}$. But it can be defined by a computable infnite disjunction of them.

Example II.1.6. On the structure $\mathcal{Q} = (\mathbb{Q}; 0, 1, +, \times)$, a relation $R \subseteq \mathbb{Q}^{<\mathbb{N}}$ is r.i.c.e. if and only if it is c.e. This is because if $\mathcal{A}$ is a copy of $\mathcal{Q}$, then there is a $D(\mathcal{A})$ computable isomorphism between $\mathcal{A}$ and $\mathcal{Q}$, and hence if R is c.e., $R^{\mathcal{A}}$ is c.e. in $D(\mathcal{A})$.

Observation II.1.7. For the definition of r.i.c.e., it does not matter whether we use ω-presentations or congruence $(\subseteq \omega)$-presentations. That is, a relation $R \subseteq A^{<\mathbb{N}}$ is r.i.c.e. as in Definition II.1.3 if and only if, for every congruence $(\subseteq \omega)$-presentation $(\mathcal{B}, R^{\mathcal{B}})$ of $(\mathcal{A}, R)$, we have that $R^{\mathcal{B}}$ is c.e. in $D(\mathcal{B})$.

II.1.2. R.i. computability. The same idea from the definition of r.i.c.e. can be used to define other standard concepts from computability theory on the subsets of $A^{<\mathbb{N}}$.

Definition II.1.8. A relation $R \subseteq A^{<\mathbb{N}}$ is *relatively intrinsically computable*(*r.i. computable*) if $R^{\mathcal{B}}$ is computable in $D(\mathcal{B})$ whenever $(\mathcal{B}, R^{\mathcal{B}})$ is a copy of $(\mathcal{A}, R)$.

Observe that R is r.i. computable if and only if it is r.i.c.e. and co-r.i.c.e. The reader can imagine how to continue in this line of definitions for other notions of complexity, like *relatively intrinsically* Δ^0_2, *relatively intrinsically arithmetic*, etc. These notions relativize in an obvious way to produce a notion of relative computability:

DEFINITION II.1.9. Given $R \subseteq A^{<\mathbb{N}}$ and $Q \subseteq A^{<\mathbb{N}}$, we say that R *is r.i.c.e. in* Q if R is r.i.c.e. in the structure $(\mathcal{A}, Q)$. R is *r.i. computable in* Q, and we write $R \leq_{rT}^{\mathcal{A}} Q$, if R is r.i. computable in the structure $(\mathcal{A}, Q)$.

The 'rT' stands for "relatively Turing." Unless we need to highlight the underlying structure, we will write $\leq_{rT}$ instead of $\leq_{rT}^{\mathcal{A}}$.

EXAMPLE II.1.10. Let $\mathcal{A} = (A; \leq)$ be a linear ordering, and consider the relation given by the pairs of elements which have at least two elements in between:

$$T = \{(a, b) \in A^2 : a < b \wedge \exists c, d(a < c < d < b)\}.$$

Then $T \leq_{rT}$ Adj: Suppose we are given $(a, b) \in A^2$ with $a < b$ and we want to decide if $(a, b) \in T$ using Adj. If $\mathrm{Adj}(a, b)$, we know $(a, b) \notin T$. Otherwise, search for c in between a and b, which we know we will find. Then we have that $(a, b) \in T$ if an only if either $\neg\mathrm{Adj}(a, c)$ or $\neg\mathrm{Adj}(c, b)$.

On the linear ordering of the natural numbers $\omega = (\mathbb{N}; \leq)$, we also have Adj $\leq_{rT} T$: To decide if a and b are adjacent wait either for an element to appear in between them or for an element $c > b$ with $\neg T(a, c)$. We have $\neg\mathrm{Adj}(a, b)$ in the former case and $\mathrm{Adj}(a, b)$ in the latter.

On the other hand, there are linear orderings where Adj $\not\leq_{rT} T$. As an example, consider the linear ordering

$$\mathcal{A} = 2\mathbb{Q} + 3 + 2\mathbb{Q} + 3 + \ldots . $$

To show that Adj $\not\leq_{rT} T$, it is enough to build a computable copy $\mathcal{B}$ of $\mathcal{A}$, where $T^{\mathcal{B}}$ is computable, but $\mathrm{Adj}^{\mathcal{B}}$ is not. To do this, let us start by considering a computable ω-presentation $\mathcal{C}$ of the linear ordering $2\mathbb{Q}$, and picking a computable increasing sequence of adjacent pairs $c_{n,0}, c_{n,1}$ for $n \in \mathbb{N}$. To build the ω-presentation $\mathcal{B}$ of $\mathcal{A}$, we will add an element in between $c_{n,0}$ and $c_{n,1}$ if and only if $n \in 0'$; we can then decode $0'$ from $\mathrm{Adj}^{\mathcal{B}}$ by checking if $c_{n,0}$ and $c_{n,1}$ are adjacent in $\mathcal{B}$. More formally, to define $\mathcal{B}$, put a copy of $\mathcal{C}$ on the even numbers in the domain of $\mathcal{B}$, and use the odd numbers to add those "in-between" elements. Let $2s + 1$ be $\leq_{\mathcal{B}}$-between $c_{k_s,0}$ and $c_{k_s,1}$, where $\{k_s : s \in \mathbb{N}\}$ is a computable enumeration of $0'$. Notice that $\leq_{\mathcal{B}}$ is computable. The relation $T^{\mathcal{B}}$ is also computable, as it holds between any two elements of $\mathcal{C}$ which are not in the same 2-block, and holds between $2s + 1$ and any other element, except for $c_{k_s,0}$ and $c_{k_s,1}$.

II.1.3. A syntactic characterization. R.i.c.e. relations can be characterized in a purely syntactic way using computably infinitary formulas and without referring to the different copies of the structure. We will define computably infinitary formulas in Part 2. For now, only define the class of *computably infinitary* Σ_1 *formulas* or $\Sigma_1^{\mathtt{c}}$ *formulas*.

DEFINITION II.1.11. An *infinitary* Σ_1 *formula* (denoted $\Sigma_1^{\mathtt{in}}$) is a countable infinite (or finite) disjunction of $\exists$-formulas over a finite set of free variables. A *computable infinitary* Σ_1 *formula* (denoted $\Sigma_1^{\mathtt{c}}$) is an infinite or finite disjunction of a computable list of $\exists$-formulas over a finite set of free variables.

Thus, a $\Sigma_1^{\mathtt{c}}$ formula is one of the form

$$\psi(\bar{x}) \quad \equiv \quad \bigvee_{i \in I} \exists \bar{y}_i \varphi_i(\bar{x}, \bar{y}_i),$$

where each φ_i is quantifier-free, I is an initial segment of $\mathbb{N}$, and the list of Gödel indices $\langle \ulcorner \varphi_i \urcorner : i \in \mathbb{N} \rangle$ is c.e. The definition of "$\mathcal{A} \models \psi(\bar{a})$" is straightforward. Using the effective enumeration $\{W_e : e \in \mathbb{N}\}$ of the c.e. sets, we can easily enumerate all Σ^{c}_1 formulas as follows: If $\{\varphi^{\exists}_{n,j}(x_1, ..., x_j) : n \in \mathbb{N}\}$ is an effective enumeration of the existential τ-formulas with j free variables, we define

$$\varphi^{\Sigma^{\mathsf{c}}_1}_{e,j}(\bar{x}) \quad \equiv \quad \bigvee_{\langle n,j \rangle \in W_e} \varphi^{\exists}_{n,j}(\bar{x})$$

for each $e \in \mathbb{N}$. We then get that $\{\varphi^{\Sigma^{\mathsf{c}}_1}_{e,j} : e \in \mathbb{N}\}$ is an effective enumeration of the Σ^{c}_1 τ-formulas with j free variables. Note that if $\psi(\bar{x})$ is Σ^{c}_1, then $\{\bar{a} \in A^{|\bar{x}|} : \mathcal{A} \models \psi(\bar{a})\}$ is c.e. in $D(\mathcal{A})$, uniformly in ψ and $\mathcal{A}$. In other words, there is a c.e. operator W such that $(\ulcorner \psi \urcorner, \bar{a}) \in W^{D(\mathcal{A})}$ if and only if $\mathcal{A} \models \psi(\bar{a})$ for all τ-structures $\mathcal{A}$ and Σ^{c}_1-τ-formulas ψ.

Example II.1.12. In a group $\mathcal{G} = (G; e, *)$, the set of torsion elements can be described by the Σ^{c}_1 formula:

$$\text{torsion}(x) \quad \equiv \quad \bigvee_{i \in \mathbb{N}} \left(\underbrace{x * x * \cdots * x}_{i \text{ times}} = e \right).$$

On a graph $\mathcal{G} = (V; E)$, the relation of being path-connected can be described by the Σ^{c}_1 formula:

$$\text{connected}(x, y) \quad \equiv \quad \bigvee_{i \in \mathbb{N}} \exists z_1, ..., z_i \, (xEz_1 \wedge z_1Ez_2 \wedge \cdots \wedge z_iEy).$$

We would like to consider Σ^{c}_1-definability, not only for n-ary relations, but also for subsets of $A^{<\mathbb{N}}$.

Definition II.1.13. A relation $R \subset A^{<\mathbb{N}}$ is Σ^{c}_1-*definable* in $\mathcal{A}$ with parameters if there is a tuple $\bar{p} \in \mathcal{A}^{<\mathbb{N}}$ and a computable sequence of Σ^{c}_1 formulas $\psi_i(x_1, ..., x_{|\bar{p}|}, y_1, ..., y_i)$, for $i \in \mathbb{N}$, such that

$$R = \{\bar{b} \in \mathcal{A}^{<\mathbb{N}} : \mathcal{A} \models \psi_{|\bar{b}|}(\bar{p}, \bar{b})\}.$$

(When we say "computable sequence of Σ^{c}_1 formulas," we of course mean of indices of Σ^{c}_1 formulas.)

From the observation before Example II.1.12, it is not hard to see that if $R \subset A^{<\mathbb{N}}$ is Σ^{c}_1 definable in $\mathcal{A}$ with parameters, it is r.i.c.e. The next theorem shows that this is a characterization. The theorem was proved for n-ary relations by Ash, Knight, Manasse, and Slaman [**AKMS89**], and by Chisholm [**Chi90**] independently. The proof for subsets of $A^{<\mathbb{N}}$ is no different.

Theorem II.1.14 (Ash, Knight, Manasse, Slaman [**AKMS89**]; Chisholm [**Chi90**]). *Let $\mathcal{A}$ be a structure, and $R \subseteq A^{<\mathbb{N}}$ a relation on it. The following are equivalent:*

(1) *R is r.i.c.e.*
(2) *R is Σ^{c}_1 definable in $\mathcal{A}$ with parameters.*

Proof. As we mentioned above, (2) easily implies (1). We prove the other direction. We will build a copy $\mathcal{B}$ of $\mathcal{A}$ by taking the pull-back of an enumeration $g \colon \mathbb{N} \to A$ that

we construct step by step, and we will apply (1) to that copy. To define $\mathcal{B}$, we define g as the union of a nested sequence of tuples $\{\bar{p}_s : s \in \mathbb{N}\} \subseteq A^{<\mathbb{N}}$, where $\bar{p}_s$ is defined at stage s. Then we define $\mathcal{B}$ to be the pull-back $g^{-1}(\mathcal{A})$ as in Subsection I.1.7. Thus, we will have

$$\bar{p}_0 \subseteq \bar{p}_1 \subseteq \bar{p}_2 \subseteq \cdots \xrightarrow{s\to\infty} g \quad \text{and} \quad D(\mathcal{B}) = \bigcup_s D_{\mathcal{A}}(\bar{p}_s).$$

Throughout the construction, we try as much as possible to make $R^{\mathcal{B}}$ not c.e. in $D(\mathcal{B})$. But, because of (1), this attempt will fail somewhere, and we will have that

$$g^{-1}(R) = R^{\mathcal{B}} = W_e^{D(\mathcal{B})}$$

for some $e \in \mathbb{N}$. We will then turn this failure into a $\Sigma^{\mathfrak{c}}_1$ definition of R.

Here is the construction of $\mathcal{B}$. Let $\bar{p}_0$ be the empty sequence. At odd stages, we take one step towards making g onto. At stage $s+1 = 2e+1$, if the eth element of A is not already in $\bar{p}_s$, we add it to the range of $\bar{p}_{s+1}$ (i.e., we let $\bar{p}_{s+1} = \bar{p}_s{}^\frown e$), and otherwise let $\bar{p}_{s+1} = \bar{p}_s$.

At the even stages, we work towards making $R^{\mathcal{B}}$ not c.e. in $D(\mathcal{B})$: At stage $s+1 = 2e$, we try to force $W_e^{D(\mathcal{B})} \not\subseteq g^{-1}(R)$ by trying to get $\langle j_1, ..., j_\ell\rangle \in W_e^{D(\mathcal{B})}$ while $\langle g(j_1), ..., g(j_\ell)\rangle \notin R$ for some tuple $j_1, ..., j_\ell \in \mathbb{N}$. We do this as follows: We search for an extension $\bar{q}$ of $\bar{p}_s$ in the set

$$Q_e = \{\bar{q} \in A^{<\mathbb{N}} : \exists \ell, j_1, ..., j_\ell < |\bar{q}| \ \big(\langle j_1, ..., j_\ell\rangle \in W_e^{D_{\mathcal{A}}(\bar{q})} \text{ and } \langle q_{j_1}, ..., q_{j_\ell}\rangle \notin R\big)\}.$$

If we find one, we let $\bar{p}_{s+1} = \bar{q}$. If not, we do nothing and let $\bar{p}_{s+1} = \bar{p}_s$. This ends the construction of g and $\mathcal{B}$.

$$\begin{array}{ccccc} B^{<\omega} & = & \mathbb{N}^{<\omega} & \xrightarrow{\quad g \quad} & A^{<\omega} \\ & & \cup\!| & & \cup\!| \\ & & W_e^{D(\mathcal{B})} & & R \\ & & \cup & & \curlywedge \\ & & \langle j_1, ..., j_\ell\rangle & \xrightarrow{\bar{q}} & \langle q_{j_1}, ..., q_{j_\ell}\rangle \end{array}$$

Notice that if at a stage $s+1 = 2e$, we succeed in defining $\bar{p}_{s+1} = \bar{q} \in Q_e$, then we succeed in making $W_e^{D(\mathcal{B})} \neq g^{-1}(R)$: This is because we would have $\bar{q} \subseteq g$ and hence that

$$\langle j_1, ..., j_\ell\rangle \in W_e^{D_{\mathcal{A}}(\bar{q})} \subseteq W_e^{D(\mathcal{B})} \quad \text{while} \quad \langle g(j_1), ..., g(j_\ell)\rangle = \langle q_{j_1}, ..., q_{j_\ell}\rangle \notin R.$$

(Observe that since $D(\mathcal{B}) = \bigcup_s D_{\mathcal{A}}(\bar{p}_s)$, we have that $W_e^{D(\mathcal{B})} = \bigcup_s W_e^{D_{\mathcal{A}}(\bar{p}_s)}$.) However, we cannot succeed at all such stages because $R^{\mathcal{B}} = W_e^{D(\mathcal{B})}$ for some $e \in \mathbb{N}$. Thus, for that particular e, at stage $s+1 = 2e$, there was no extension of $\bar{p}_s$ in Q_e.

Claim II.1.14.1. If $R^{\mathcal{B}} = W_e^{D(\mathcal{B})}$ and there are no extensions of $\bar{p}$ in Q_e, then R is $\Sigma^{\mathfrak{c}}_1$-definable in $\mathcal{A}$ with parameters $\bar{p}$.

Proof of the claim. Notice that if we find some $\bar{q} \supseteq \bar{p}$ and a sub-tuple $\bar{a} = \langle q_{j_1}, ..., q_{j_\ell}\rangle$ such that $\langle j_1, ..., j_\ell\rangle \in W_e^{D_{\mathcal{A}}(\bar{q})}$, then we must have $\bar{a} \in R$, as otherwise we would get $\bar{q} \in Q_e$. This is the key idea we use to enumerate elements into R.

More formally, we will show that R is equal to the set

$$S = \{\langle q_{j_1}, ..., q_{j_\ell}\rangle \in A^{<\omega} : \text{ where } \bar{q} \in A^{<\mathbb{N}} \text{ and } \ell, j_1, ..., j_\ell < |\bar{q}| \\ \text{satisfying } \bar{q} \supseteq \bar{p} \text{ and } \langle j_1, ..., j_\ell\rangle \in W_e^{D_{\mathcal{A}}(\bar{q})}\}.$$

If $\bar{a} \in R$, let $j_1, ..., j_{|\bar{a}|}$ be indices such that $\bar{a} = \langle g(j_1), ..., g(j_{|\bar{a}|})\rangle$, and we get that $\bar{a} \in S$ using for $\bar{q}$ a long enough segment of g. For the other direction, if $\bar{a} = \langle q_{j_1}, ..., q_{j_\ell}\rangle \in S$, then we must have $\bar{a} \in R$, as otherwise we would have $\bar{q} \in Q_e$, contradicting the assumption of the claim.

Now that we know that $R = S$, let us show that S is $\Sigma_1^{\mathfrak{c}}$ definable with parameters $\bar{p}$. For every $\alpha \in A^{<\omega}$,

$$\bar{a} \in S \iff \exists \bar{q} \supseteq \bar{p} \bigvee_{j_1,...,j_{|\bar{a}|} < |\bar{q}|} \left(\langle q_{j_1}, ..., q_{j_{|\bar{a}|}}\rangle = \bar{a} \quad \& \quad \langle j_1, ..., j_{|\bar{a}|}\rangle \in W_e^{D_{\mathcal{A}}(\bar{q})} \right).$$

But "$\langle j_1, ..., j_{|\bar{a}|}\rangle \in W_e^{D_{\mathcal{A}}(\bar{q})}$" is not a formula in the language. So, we need to re-write it as:

$$\bar{a} \in S \iff \bigvee_{\sigma \in 2^{<\mathbb{N}},\, (j_1,...,j_{|\bar{a}|}) \in W_e^\sigma} \bigvee \exists \bar{q} \supseteq \bar{p} \ \left(\langle q_{j_1}, ..., q_{j_{|\bar{a}|}}\rangle = \bar{a} \ \& \ \text{“}\sigma \subseteq D_{\mathcal{A}}(\bar{q})\text{”} \right).$$

Recall that, for each $\sigma \in 2^{<\mathbb{N}}$, there is a quantifier-free formula with the meaning "$\sigma \subseteq D_{\mathcal{A}}(\bar{x})$" (Observation I.1.8). □

Thus, R is $\Sigma_1^{\mathfrak{c}}$-definable in $\mathcal{A}$ with parameters $\bar{p}_s$.

(For the detail-oriented reader, let us observe that the fact that the congruence ω-presentation $\mathcal{B}$ is non-injective is not important here by Observation II.1.7.) □

Very often, we will deal with relations that are $\Sigma_1^{\mathfrak{c}}$-definable without parameters. These relations are not just r.i.c.e., but uniformly r.i.c.e.:

Definition II.1.15. A relation $R \subseteq A^{<\mathbb{N}}$ is uniformly r.i.c.e. (u.r.i.c.e.) if there is a c.e. operator W such that $R^{\mathcal{B}} = W^{D(\mathcal{B})}$ for all $(\mathcal{B}, R^{\mathcal{B}}) \cong (\mathcal{A}, R)$.

The difference between r.i.c.e. and u.r.i.c.e. relations is just that the former needs parameters in its $\Sigma_1^{\mathfrak{c}}$-definition — parameters that one may no be able to find computably and hence require "non-uniform" information.

Corollary II.1.16. *Let $\mathcal{A}$ be a structure and $R \subseteq A^{<\mathbb{N}}$ a relation on it. The following are equivalent:*

1. *R is u.r.i.c.e.*
2. *R is $\Sigma_1^{\mathfrak{c}}$ definable in $\mathcal{A}$ without parameters.*

Proof. It is easy to see that (2) implies (1). For the other direction, let W_e be the c.e. operator witnessing that R is u.r.i.c.e.. Let Q_e be as in the proof of Theorem II.1.14. No tuple $\bar{q} \in A^{<\mathbb{N}}$ can be in Q_e because, otherwise, any extension of $\bar{q}$ to an enumeration g of $\mathcal{A}$ would satisfy $W_e^{D(g^{-1}(\mathcal{A}))} \not\subseteq g^{-1}(R)$, contradicting our choice of W_e. The corollary then follows from Claim II.1.14.1. □

II.1.4. Coding sets of natural numbers. Another feature that is useful when working with subsets of $\mathcal{A}^{<\mathbb{N}}$ is that we can code subsets of $\mathbb{N}$ in a straightforward way:

Definition II.1.17. Given a set $X \subseteq \mathbb{N}$, we define $\vec{X} \subseteq \mathcal{A}^{<\mathbb{N}}$ by letting $\bar{b} \in \vec{X}$ if and only if $|\bar{b}| \in X$.

When $\vec{X}$ is r.i.c.e. in $\mathcal{A}$, we say that X is *coded by* $\mathcal{A}$ [**Mon10**, Definition 1,8]. It follows from the definitions that $\vec{X}$ is r.i.c.e. in $\mathcal{A}$ if and only if X is c.e. in every ω-presentation of $\mathcal{A}$. A characterization of the sets X coded by a given structure was first given by Knight [**Kni86**, Theorem 1.4'], and we get it as a corollary of Theorem II.1.14. Let us first see a couple of examples.

Example II.1.18. Given $X \subseteq \mathbb{N}$, let $\mathcal{G}$ be the group $\bigoplus_{i \in X} \mathbb{Z}_{p_i}$, where p_i is the ith prime number. We then have that X is coded by $\mathcal{G}$, as $i \in X$ if and only if there is an element of $\mathcal{G}$ with order p_i.

A more general family of examples are the $\exists$-types of tuples from the structure.

Definition II.1.19. Given $\bar{a} \in A^{<\mathbb{N}}$, we define the $\exists$-*type* of $\bar{a}$ in $\mathcal{A}$ as

$$\exists\text{-}tp_{\mathcal{A}}(\bar{a}) = \{i \in \mathbb{N} : \mathcal{A} \models \varphi^{\exists}_{i,|\bar{a}|}(\bar{a})\}$$

where $\langle \varphi^{\exists}_{i,j} : i \in \mathbb{N} \rangle$ is an effective enumeration of the $\exists$-τ-formulas with j-free variables.

Clearly, for any tuple $\bar{a} \in A^{<\mathbb{N}}$, we can enumerate $\exists$-$tp_{\mathcal{A}}(\bar{a})$ from any ω-presentation of $\mathcal{A}$ once we recognize where the tuple $\bar{a}$ is in the ω-presentation (non-uniformly). Knight's theorem essentially says that $\exists$-types are essentially all that a structure can code. To state Knight's results, we need to introduce the notion of *enumeration reducibility*.

Definition II.1.20. An *enumeration of* Y is an onto function $f \colon \mathbb{N} \to Y$. A set $X \subseteq \mathbb{N}$ is *e-reducible to* $Y \subseteq \mathbb{N}$ if every enumeration of Y computes an enumeration of X. See the background section for more on e-reducibility.

If a set $X \subseteq$ is e-reducible to the $\exists$-type of some tuple $\bar{a}$ in $\mathcal{A}$, then any ω-presentation of $\mathcal{A}$ can enumerate $\exists$-$tp_{\mathcal{A}}(\bar{a})$ and hence also X. It follows that X is coded by $\mathcal{A}$. Knight showed that these are all the sets $\mathcal{A}$ codes:

Corollary II.1.21 (Knight [**Kni86**, Theorem 1.4'], see also [**AK00**, Theorem 10.17]). *Let $X \subseteq \mathbb{N}$. The following are equivalent:*

(B1) *X is coded by $\mathcal{A}$ (i.e., X is c.e. in every copy of $\mathcal{A}$).*
(B2) *X is e-reducible to $\exists$-$tp_{\mathcal{A}}(\bar{p})$ for some $\bar{p} \in A^{<\mathbb{N}}$.*

Proof. We have already mentioned how (B2) implies (B1). We prove the other direction.

As we mentioned before, X is c.e. in every copy of $\mathcal{A}$ if and only if $\vec{X}$ is r.i.c.e. in $\mathcal{A}$. By Theorem II.1.14, we have a Σ^{c}_1 definition of $\vec{X}$ over some parameters $\bar{p}$. We can then transform this Σ^{c}_1 definition into an enumeration operator Φ that outputs X, given $\exists$-$tp_{\mathcal{A}}(\bar{p})$ as input: The operator Φ enumerates n into $\Phi^{\exists\text{-}tp_{\mathcal{A}}(\bar{p})}$ if (the index of) one of the disjuncts that appears in the Σ^{c}_1 definition of $\vec{X} \cap A^n$ appears in $\exists$-$tp_{\mathcal{A}}(\bar{p})$. More formally, the Σ^{c}_1-definition of $\vec{X}$ is of the form

$$\bar{b} \in \vec{X} \cap A^n \iff \bigvee \{\varphi^{\exists}_{i,|\bar{p}|+|\bar{b}|}(\bar{p}, \bar{b}) : \langle i, |\bar{p}| + |\bar{b}| \rangle \in W\}$$

for some c.e. set W. Then

$$n \in X \iff \exists i \in \mathbb{N}\big(\langle i, |\bar{p}| + n\rangle \in W \wedge \ulcorner \exists \bar{x} \varphi_{i,|\bar{p}|+n}(\bar{p}, \bar{x})\urcorner \in \exists\text{-}tp_{\mathcal{A}}(\bar{p})\big),$$

and hence X is c.e. in every ω-presentation of $\mathcal{A}$. □

II.1.5. Joins. The use of subsets of $A^{<\mathbb{N}}$ not only allows us to consider all n-tuples simultaneously and to consider sets of natural numbers, but also all finite objects that can be built over $\mathcal{A}$. We will see more on this in Section II.4. For now, we see how to code many relations using just one.

Definition II.1.22. Given $R, Q \subseteq A^{<\mathbb{N}}$, we define $R \oplus Q$ by $\bar{b} \in R \oplus Q$ if either $|\bar{b}| = 2n$ and $\bar{b} \upharpoonright n \in R$, or $|\bar{b}| = 2n + 1$ and $\bar{b} \upharpoonright n \in Q$.

It is not hard to see that $\oplus$ defines a least-upper-bound operation for r.i. computability. That is, R and Q are r.i. computable in $R \oplus Q$, and whenever both R and Q are r.i. computable in a relation $S \subseteq A^{<\mathbb{N}}$, $R \oplus Q$ is r.i. computable in S too.

Furthermore, we can take joins of $\mathbb{N}$-sequences of relations.

Definition II.1.23. Given $Q \subseteq \mathbb{N} \times A^{<\mathbb{N}}$, define $R \subseteq A^{<\mathbb{N}}$ as follows: $\bar{b} \in R$ if and only if $|\bar{b}| = \langle n, m\rangle$ for some $n, m \in \mathbb{N}$ and $(n, \bar{b} \upharpoonright m) \in Q$.

We then have that Q is r.i.c.e. in the sense that $Q^{\mathcal{B}} \subseteq \mathbb{N} \times \mathbb{N}^{<\mathbb{N}}$ is c.e. in $\mathcal{B}$ for every copy $(\mathcal{B}, Q^{\mathcal{B}})$ of $(\mathcal{A}, Q)$ if and only if R is r.i.c.e. We can keep on pushing this idea much further. For instance, given $Q \subseteq (A^{<\mathbb{N}})^2$, define $R \subseteq \mathcal{A}^{<\mathbb{N}}$ as follows: $\bar{b} \in R$ if $|\bar{b}| = \langle n, m\rangle$ for some $n, m \in \mathbb{N}$ and $((b_0, ..., b_{n-1}), (b_n, ..., b_{n+m-1})) \in Q$, (assuming the coding of pairs satisfies $\langle n, m\rangle < n + m$). In a similar way, the reader can imagine how to code subsets of $(\mathcal{A}^{<\mathbb{N}})^{<\mathbb{N}}$ by subsets of $\mathcal{A}^{<\mathbb{N}}$. We will see the most general form of this in Section II.4.1.

II.2. Complete relations

So far we have notions of c.e.-ness, computability, and join on the subsets of $A^{<\mathbb{N}}$. The next step is to get an analog for the Turing jump.

II.2.1. R.i.c.e. complete relations.

Definition II.2.1. A relation $R \subseteq A^{<\mathbb{N}}$ is *complete* in $\mathcal{A}$ if every r.i.c.e. relation $Q \subseteq A^{<\mathbb{N}}$ is r.i. computable from R. R is *r.i.c.e. complete* if it is also r.i.c.e. itself.

The relation $\vec{0'}$, coding $0'$ as in Definition II.1.17, is always r.i.c.e. and hence every complete relation must r.i. compute it. In some cases, $\vec{0'}$ is complete itself, but in most cases, it is not. This is not surprising, because $\vec{0'}$ contains no structural information about $\mathcal{A}$, so one should not expect that it is going to r.i. compute all other r.i.c.e. relations.

Example II.2.2. On a $\mathbb{Q}$-vector space, $LD \oplus \vec{0'}$ is r.i.c.e. complete, but LD is not r.i. computable from $\vec{0'}$ when the space has infinite dimension. Recall that LD is the linear dependence relation. On a linear ordering, $(\neg\text{Adj}) \oplus \vec{0'}$ is r.i.c.e. complete, but $\neg$Adj is not r.i. computable from $\vec{0'}$ unless there are only finitely many adjacencies. We will prove these facts in Lemmas II.3.1 and II.3.4.

These examples of complete relations are particularly nice and clean, but we will not always be able to find such simple complete relations. Simple or not, r.i.c.e. complete relations always exist. For instance, we can consider the analog of Kleene's predicate K by putting together all $\Sigma^{\mathfrak{c}}_1$-definable relations.

DEFINITION II.2.3. [**Mon12**] The *Kleene relation* relative to $\mathcal{A}$, denoted $\vec{K}^{\mathcal{A}} \subseteq \mathbb{N} \times A^{<\mathbb{N}}$, is defined by

$$\langle i, \bar{b} \rangle \in \vec{K}^{\mathcal{A}} \iff \mathcal{A} \models \varphi^{\Sigma^{\mathfrak{c}}_1}_{i,|\bar{b}|}(\bar{b}),$$

where $\{\varphi^{\Sigma^{\mathfrak{c}}_1}_{i,j} : i \in \mathbb{N}\}$ is an effective enumeration of the $\Sigma^{\mathfrak{c}}_1$-τ-formulas with j variables as in Section II.1.3. Of course, we can code $\vec{K}^{\mathcal{A}}$ by a subset of $A^{<\mathbb{N}}$ as in Definition II.1.23.

REMARK II.2.4. It is clear that $\vec{K}^{\mathcal{A}}$ is r.i.c.e. To show that it is complete, it follows from Theorem II.1.14 that, for every r.i.c.e. $R \subseteq A^{<\mathbb{N}}$, there is a tuple $\bar{a}$ and a computable function $n \mapsto e_n$ such that

$$\bar{b} \in R \iff \mathcal{A} \models \varphi^{\Sigma^{\mathfrak{c}}_1}_{e_n,|\bar{a}|+n}(\bar{a}, \bar{b}) \qquad \text{for all } n \in \mathbb{N} \text{ and } \bar{b} \in A^n.$$

One can then show that the right-hand-side can we written as a question of the form $\langle e, \bar{a}\bar{b} \rangle \in \vec{K}^{\mathcal{A}}$. For the reader interested in the details, here is the proof. Recall that $\varphi^{\Sigma^{\mathfrak{c}}_1}_{e,j}(\bar{x})$ was defined as $\bigvee_{\langle i,j \rangle \in W_e} \varphi^{\exists}_{i,j}(\bar{x})$ where $j = |\bar{x}|$. Let e be the index for W_e such that

$$\langle i, |\bar{a}| + n \rangle \in W_e \iff \langle i, |\bar{a}| + n \rangle \in W_{e_n}.$$

Then, we get $(\forall n \in \mathbb{N})\ \varphi^{\Sigma^{\mathfrak{c}}_1}_{e,|\bar{a}|+n} \equiv \varphi^{\Sigma^{\mathfrak{c}}_1}_{e_n,|\bar{a}|+n}$, and that

$$\bar{b} \in R \iff \langle e, \bar{a}\bar{b} \rangle \in \vec{K}^{\mathcal{A}} \qquad \text{for all } \bar{b} \in A^{<\mathbb{N}}.$$

In particular, it follows that, given an enumeration of all tuples in $\mathcal{A}$, we can get an enumeration of all r.i.c.e. subsets of $A^{<\mathbb{N}}$.

EXERCISE II.2.5. Generalize Remark II.2.4 and get that not only is $\vec{K}^{\mathcal{A}}$ complete among r.i.c.e. subsets of $\mathcal{A}^{<\mathbb{N}}$, but also among subsets of $\mathbb{N} \times \mathcal{A}^{<\mathbb{N}}$: For every r.i.c.e. $\vec{R} \subseteq \mathbb{N} \times A^{<\mathbb{N}}$, there is a tuple $\bar{a}$ and a computable function $f \colon \mathbb{N} \to \mathbb{N}$ such that

$$\langle i, \bar{b} \rangle \in \vec{R} \iff \langle f(i), \bar{a}\bar{b} \rangle \in \vec{K}^{\mathcal{A}} \qquad \text{for all } i \in \mathbb{N} \text{ and } \bar{b} \in A^{<\mathbb{N}}.$$

In terms of Turing degree, it is easy to see that $\vec{K}^{\mathcal{A}} \leq_T D(\mathcal{A})'$ for any ω-presentation $\mathcal{A}$. The reverse reducibility holds in some ω-presentations (Lemma IV.3.4) and not in others:

EXERCISE II.2.6. Show that any non-trivial structure has an ω-presentation $\mathcal{A}$ with $D(\mathcal{A}) \equiv_T \vec{K}^{\mathcal{A}}$.

By relativizing Kleene's relation, we can define a jump operator on subsets of $A^{<\mathbb{N}}$.

DEFINITION II.2.7. Given $Q \in A^{<\mathbb{N}}$, we define *the jump of Q in $\mathcal{A}$* to be $\vec{K}^{(\mathcal{A},Q)}$, that is, Kleene's relation as in Definition II.2.3 relative to the structure $(\mathcal{A}, Q)$. We denote it by Q'.

II.2.2. Diagonalization. We now prove that, on the space of subsets of $A^{<\mathbb{N}}$, the jump operation actually jumps.

THEOREM II.2.8. *For every structure $\mathcal{A}$, $\vec{K}^{\mathcal{A}}$ is not r.i. computable in $\mathcal{A}$.*

PROOF. This proof is essentially the same as Kleene's diagonalization argument for showing that $0'$ is not computable, but adapted to this setting.

Consider the following modification of $\vec{K}^{\mathcal{A}}$:

$$\vec{K}_2^{\mathcal{A}} = \{\langle\langle e, \bar{a}\rangle, \langle i, \bar{b}\rangle\rangle \in (\mathbb{N} \times A^{<\omega})^2 : \Phi_e(i)\downarrow \quad \text{and} \quad (\langle \Phi_e(i), \bar{a}\bar{b}\rangle \notin \vec{K}^{\mathcal{A}}\}.$$

By the completeness of $\vec{K}^{\mathcal{A}}$ as in Exercise II.2.5, we have that for every r.i.c.e. relation $\vec{R} \subseteq \mathbb{N} \times A^{<\omega}$, there is an e and an $\bar{a}$ such that $\vec{R}$ is the $\langle e, \bar{a}\rangle$-th column of $\vec{K}_2^{\mathcal{A}}$, that is, $\langle i, \bar{b}\rangle \in \vec{R} \iff \langle\langle e, \bar{a}\rangle, \langle i, \bar{b}\rangle\rangle \in \vec{K}_2^{\mathcal{A}}$.

Consider now the complement of the diagonal of $\vec{K}_2^{\mathcal{A}}$:

$$\vec{R} = \{\langle e, \bar{b}\rangle \in \mathbb{N} \times A^{<\mathbb{N}} : \quad \Phi_e(e)\downarrow \quad \text{and} \quad \langle \Phi_e(e), \bar{b}\bar{b}\rangle \notin \vec{K}^{\mathcal{A}}\}.$$

Suppose, toward a contradiction, that $\vec{K}^{\mathcal{A}}$ is co-r.i.c.e. We then get that $\vec{R}$ is r.i.c.e. too. By the completeness of $\vec{K}_2^{\mathcal{A}}$, $\vec{R}$ must be one of the columns of $\vec{K}_2^{\mathcal{A}}$. That is, there is an $\bar{a} \in A^n$ and an index k for a total computable function $f = \Phi_k$ such that

$$\langle i, \bar{b}\rangle \in \vec{R} \iff \langle \Phi_k(i), \bar{a}\bar{b}\rangle \in \vec{K}^{\mathcal{A}} \qquad \text{for all } \langle i, \bar{b}\rangle \in \mathbb{N} \times A^{<\mathbb{N}}.$$

We then get the following contradiction:

$$\langle k, \bar{a}\rangle \in \vec{R} \iff \langle \Phi_k(k), \bar{a}\bar{a}\rangle \in \vec{K}^{\mathcal{A}} \iff \langle k, \bar{a}\rangle \notin \vec{R}. \qquad \square$$

COROLLARY II.2.9. *For every $Q \in A^{<\mathbb{N}}$, $Q <_{rT}^{\mathcal{A}} Q'$; that is, Q is r.i. computable in Q', but Q' is not r.i. computable in Q.*

PROOF. It is easy to see that $Q \leq_{rT}^{\mathcal{A}} Q'$ because the $\Sigma_1^{\mathfrak{c}}$ diagram of $(\mathcal{A}, Q)$ clearly computes the atomic diagram of $(\mathcal{A}, Q)$ in any copy of $\mathcal{A}$. That Q' is not r.i. computable in Q follows from the theorem above applied to the structure $(\mathcal{A}, Q)$. $\square$

HISTORICAL REMARK II.2.10. *The proof of Theorem II.2.8 given above is from* [**Mon12**], *although it is clearly similar to the standard proof of the incomputability of the halting problem. Theorem II.2.8 had been previously proved for a different, yet equivalent, notion of jump by Vatev in* [**Vat11**]. *Vatev's proof, restated in our terms, goes by showing that if $\mathcal{B}$ is a generic copy of $\mathcal{A}$, then $\vec{K}^{\mathcal{B}}$ has degree $D(\mathcal{B})'$ (which, of course, is not computable in $D(\mathcal{B})$), and hence $\vec{K}^{\mathcal{A}}$ is not r.i. computable in $\mathcal{A}$. From a personal communication, Stukachev has another proof which has not been translated into english yet.*

II.2.3. Structural versus binary information. As we saw in Definition II.1.17, we can code reals $X \subseteq \mathbb{N}$ with relations $\vec{X} \subset A^{<\mathbb{N}}$ in a somewhat trivial way using just the size of the tuples. There is no structural information on the relation $\vec{X}$: We then say that the information content in $\vec{X}$ is *purely binary*. On the contrary, relations like Adj on a linear ordering are *purely structural*. Relations like the r.i.c.e. complete relation on a linear ordering, $\neg\text{Adj} \oplus \vec{0'}$, are a mix of both. Sometimes, one is interested only in structural behavior, and having to deal with the binary parts of relations may obscure what one is trying to analyze. In that case, one should consider the *structural* versions of the notions from earlier in this chapter by modding out the binary information: A

relation $R \subseteq A^{<\mathbb{N}}$ is *structurally r.i.c.e.* in $\mathcal{A}$ if it is r.i.c.e. in $(\mathcal{A}, \vec{X})$ for some $X \subseteq \mathbb{N}$. R is *structurally r.i. computable* in Q within $\mathcal{A}$ if R is r.i. computable in $(A, Q, \vec{X})$ for some $X \subseteq \mathbb{N}$. We could also refer to these versions as the *boldface* versions or the *on-a-cone* versions. The one we will actually consider in this book is the following:

DEFINITION II.2.11. A relation R is *structurally complete* if $R \oplus \vec{X}$ is complete for some $X \subseteq \mathbb{N}$.

In all of our examples, the set X from the definition will turn out to be $0'$. However, we could come up with other examples where other sets X are needed.

II.3. Examples of r.i.c.e. complete relations

In this section, we consider structures which have nice structurally complete relations.

LEMMA II.3.1. *Let $\mathcal{A} = (A; \leq)$ be a linear ordering. Then*

$$Adj = \{(a, b) \in A^2 : a < b \ \& \ \nexists c \ (a < c < b)\}.$$

is structurally complete. Furthermore, $\neg Adj \oplus \vec{0'}$ is r.i.c.e. complete.

PROOF. The proof goes by showing that every Σ^{c}_1 formula is equivalent to a finitary universal formula over the vocabulary $\{\leq, \mathrm{Adj}\}$, and that $0'$ can find this equivalent formula uniformly. We then get that every Σ^{c}_1-definable relation is r.i. computable in $\mathrm{Adj} \oplus \vec{0'}$. One could prove this in a purely syntactical way in the style of a quantifier elimination argument, but instead we give a more model-theoretic proof.

Let $\varphi(x_1, ..., x_k)$ be a Σ^{c}_1 formula about linear orderings. Let $\bar{c} = (c_1, ..., c_k)$ be new constant symbols and $\tau' = \{\leq, c_1, ..., c_k\}$. We will use the term *$\bar{c}$-linear ordering* to refer to a linear ordering where the constants from $\bar{c}$ have been assigned. As a preview of the rest of the proof, let us mention that one of the key points is that the finite $\bar{c}$-linear orderings form a well-quasi-ordering under embeddability. The proof is divided into three claims:

CLAIM II.3.1.1. Two Σ^{c}_1-τ'-sentences are equivalent on $\bar{c}$-linear orderings if and only if they hold on the same finite $\bar{c}$-linear orderings.

The left-to-right direction is obvious; we prove the other direction. Let φ and ψ be two Σ^{c}_1 sentences which hold on the same finite $\bar{c}$-linear orderings. Consider an infinite $\bar{c}$-linear ordering $\mathcal{L}$ where φ holds. Then one of the $\exists$-disjuncts of φ holds in $\mathcal{L}$, and hence holds on a finite τ'-substructure of $\mathcal{L}$. By the assumption, ψ holds on that same finite $\bar{c}$-linear ordering, and by upward-persistence of Σ^{c}_1 formulas, ψ holds in $\mathcal{L}$ too.

CLAIM II.3.1.2. For every Σ^{c}_1-τ'-sentence φ, there is a finite set of finite $\bar{c}$-linear orderings such that, for any $\bar{c}$-linear ordering $\mathcal{L}$, $\mathcal{L} \models \varphi$ if and only if one of those finite $\bar{c}$-linear orderings τ'-embeds into $\mathcal{L}$. Furthermore, $0'$ can find those $\bar{c}$-linear orderings uniformly in φ.

Given a permutation $(\pi_1, ..., \pi_k)$ of $(1, ..., k)$ and $k+1$ numbers $\bar{n} = (n_0, ..., n_k)$, let $\mathcal{L}_{\pi,\bar{n}}$ be the finite $\bar{c}$-linear ordering with $c_{\pi_1} \leq c_{\pi_2} \leq \cdots \leq c_{\pi_k}$ and n_0 elements less

than c_1, n_i elements between c_i and c_{i+1}, and n_k elements greater than c_k. Consider the ordering $\preceq$ on $S_k \times \mathbb{N}^{k+1}$ given by

$$(\pi, \bar{n}) \preceq (\sigma, \bar{m}) \iff \pi = \sigma \ \& \ (\forall i \leq k)\ n_i \leq m_i,$$

where S_k is the set of permutations of $(1, ..., k)$. We then have that

$$(\pi, \bar{n}) \preceq (\sigma, \bar{m}) \ \Rightarrow \ \mathcal{L}_{\pi,\bar{n}} \ \text{ embeds in } \ \mathcal{L}_{\sigma,\bar{m}}.$$

By upward-persistence of Σ^{c}_1 formulas, it follows that the set D of $(\pi, \bar{n}) \in S_k \times \mathbb{N}^{k+1}$ such that $\mathcal{L}_{\pi,\bar{n}} \models \varphi$ is $\preceq$-upwards closed. Now, by Dickson's Lemma, the ordering $\preceq$ is a *well-quasi-ordering*. That means that every subset of $S_k \times \mathbb{N}^{k+1}$ has a finite set of minimal elements, and hence that for every upward-closed subset $D \subseteq S_k \times \mathbb{N}^{k+1}$, there is a finite set $d_1, ..., d_\ell \subseteq D$ such that

$$f \in D \iff (\exists j \leq \ell)\ d_i \preceq f \qquad \text{for all } f \in S_k \times \mathbb{N}^{k+1}.$$

The oracle $0'$ can find this finite set $\{d_1, ..., d_\ell\}$ because it can check that every $\mathcal{L}_{d_j}$ satisfies φ and that every $\mathcal{L}_f$ with $(\forall j \leq \ell)\ d_i \not\preceq f$ does not satisfy φ. This proves our second claim.

Let $\psi_n(x, y)$ be the $\exists$-formula that says that there are at least n many different elements strictly in between x and y:

$$\psi_n(x, y) \equiv \exists z_1, ..., z_n\ (x < z_1 < z_2 < \cdots < z_n < y).$$

We write $\psi_n(-\infty, y)$ for the unary $\exists$-formula that says that there are at least n many different elements less than y, and analogously with $\psi_n(x, \infty)$. Given a permutation $\pi \in S_k$ and $\bar{n} = (n_0, ..., n_k) \in \mathbb{N}^{k+1}$, we let

$$\psi_{\pi,\bar{n}}(x_1, ..., x_k) \equiv x_{\pi_1} \leq \cdots \leq x_{\pi_k} \wedge (\psi_{n_0}(-\infty, x_1) \wedge \psi_{n_1}(x_1, x_2) \wedge ... \wedge \psi_{n_k}(x_k, \infty)).$$

A $\bar{c}$-linear ordering satisfies $\psi_{\pi,\bar{n}}(c_1, ..., c_k)$ if and only if $\mathcal{L}_{\pi,\bar{n}}$ embeds in it. Then we get from the claim that every Σ^{c}_1 formula $\varphi(x_1, ..., x_k)$ is equivalent to a finite disjunction of formulas of the form $\psi_{\pi,\bar{n}}(x_{\pi_1}, ..., x_{\pi_k})$. Furthermore, $0'$ can find these formulas uniformly. The following claim is all is left to prove the lemma.

Claim II.3.1.3. The formulas $\psi_n(x, y)$ are equivalent to $\forall$-$\{\leq, \mathrm{Adj}\}$-formulas, and hence so are the formulas $\psi_{\pi,\bar{n}}(x_1, ..., x_k)$.

Just observe that $\psi_n(x, y)$ is equivalent to a finitary universal formula over the adjacency predicate:

$$\psi_n(x, y) \iff \bigwedge_{j \leq n} \nexists z_0, ..., z_j \left(x = z_0 \leq \cdots \leq z_j = y \wedge \left(\bigwedge_{i=0}^{j-1} (\mathrm{Adj}(z_i, z_{i+1}) \right) \right).$$

It follows that the relations defined by $\psi_{\pi,\bar{n}}(x_1, ..., x_k)$ are uniformly r.i. computable in Adj, and that any Σ^{c}_1 formula is uniformly r.i. computable in $\mathrm{Adj} \oplus \vec{0'}$. □

An *equivalence structure* is a structure $\mathcal{E} = (D; E)$, where E is a equivalence relation on the domain D. Define the following relations on $\mathcal{E}$:

(1) for $k \in \mathbb{N}$, $F_k = \{x \in D :$ there are $\geq k$ elements equivalent to $x\}$, and
(2) the *character* of E:
$G = \{\langle n, k\rangle \in \mathbb{N}^2 :$ there are $\geq n$ equivalence classes with $\geq k$ elements$\}$.

EXERCISE II.3.2. (a) Show that the relation $\vec{F} = \bigoplus_{k\in\mathbb{N}} F_k \subseteq \mathbb{N} \times D$ is structurally complete. (b) Show that $\vec{F} \oplus \vec{G} \oplus \vec{0'}$ is r.i.c.e. complete. Hint in footnote.[1]

EXERCISE II.3.3. Show that the "atom" relation on a Boolean algebra is structurally complete.

LEMMA II.3.4. *The relation LD of linear dependence on a $\mathbb{Q}$-vector space is structurally complete. Moreover $LD \oplus \vec{0'}$ is r.i.c.e. complete.*

PROOF. This proof is much simpler than that for Adj. The key point is that any $\mathbb{Q}$-vector space has a computable copy, and using LD, one can find an isomorphism with that computable copy because we can use LD to find a basis. If a relation is r.i.c.e., it is c.e. on the computable copy, and hence computable from $0'$ in that copy. The relation on the original ω-presentation is then computable from $LD \oplus \vec{0'}$. □

The same argument above can be used to show that the "algebraic dependence" relation is structurally complete on algebraically closed fields.

EXERCISE II.3.5. (Hard) Show that $LD_{n+1} \not\leq_{rT} LD_n$ in the ∞-dimensional $\mathbb{Q}$-vector space. Recall that LD_n is the linear dependence relation on n tuples.

II.4. Superstructures

The notion of r.i.c.e. relation is equivalent to other notions that were known many decades ago. In this section, we study one of them — the Σ-definable subsets of the hereditarily finite superstructure. There are some advantages to working in this setting: One is that r.i.c.e. relations are now defined by finitary formulas instead of computably infinitary ones. Another one is that there is almost no coding required; while subsets of $(A^{<\mathbb{N}})^{<\mathbb{N}}$ can be coded by subsets of $A^{<\mathbb{N}}$ as in Section II.1.5, subsets of $(\mathbb{HF}_{\mathcal{A}})^{<\mathbb{N}}$ are already subsets of $\mathbb{HF}_{\mathcal{A}}$. Nevertheless, the advantage of $A^{<\mathbb{N}}$ is that it is easier to visualize. At the end of the day, all these advantages, one way or the other, are purely aesthetic and not really significant.

II.4.1. The hereditarily finite superstructure. Another approach to the study of r.i.c.e. relations is using Σ-definability on admissible structures. We will not consider admissible structures in general, but just the hereditarily finite extension of an abstract structure $\mathcal{A}$. The elements of this extension are the finite sets of finite sets of $\cdots$ of finite set of elements of A.

DEFINITION II.4.1. Let $\mathcal{P}_{fin}(X)$ denote the collection of finite subsets of X. Given a set A, we define:

(1) $\mathrm{HF}_A(0) = \varnothing$,
(2) $\mathrm{HF}_A(n+1) = \mathcal{P}_{fin}(A \cup \mathrm{HF}_A(n))$, and
(3) $\mathbb{HF}_A = \bigcup_{n\in\mathbb{N}} \mathrm{HF}_A(n)$.

[1]For (b) you need to use that the set of finite subsets of $\mathbb{N}^2$ ordered by $A \leq B \iff \forall(x,y) \in A \exists(x',y') \in B\ (x \leq x' \ \&\ y \leq y')$ is well-quasi-ordered, and hence that every set has a finite subset of minimal elements.

Now, given an τ-structure $\mathcal{A}$, we define the $\tau \cup \{\in, D\}$-structure $\mathcal{HF}_\mathcal{A}$ whose domain has two sorts, A and $\mathbb{HF}_A$, and where the symbols from τ are interpreted in the A-sort as in $\mathcal{A}$, '$\in$' is interpreted in the obvious way, and D is a unary relation coding the atomic diagram of $\mathcal{A}$ defined below. The need for adding D is a slightly technical, so we will explain it later.

A quantifier of the form $\forall x \in y...$ or $\exists x \in y....$ is called a *bounded quantifier*. A Σ-*formula* is finitary $\tau \cup \{\in, D\}$-formula that is built out of atomic and negation-of-atomic formulas using disjunction, conjunction, bounded quantifiers, and existential unbounded quantifiers. A subset of $\mathcal{HF}_\mathcal{A}$ is Δ-*definable* if it and its complement are Σ-definable.

Clearly, on $\mathcal{HF}_\mathcal{A}$ we have the usual pairing function $\langle x, y\rangle = \{\{x\}, \{x, y\}\}$, and we can encode n-tuples, strings, etc. Notice also that $\mathcal{HF}_\mathcal{A}$ includes the finite ordinals (denoted by $\underline{n}$, where $\underline{0} = \varnothing$ and $\underline{n+1} = \{\underline{n}\} \cup \underline{n}$). We use ω to denote the Δ-definable set of finite ordinals of $\mathcal{HF}_\mathcal{A}$. The operations of successor, addition, and multiplication on ω are also Δ-definable, and hence so is Kleene's T predicate. It follows that every c.e. subset of ω is Σ-definable, and every computable function is Δ-definable in $\mathcal{HF}_\mathcal{A}$ (for more details, see [**Bar75**, Theorem II.2.3]).

We define $D(\mathcal{A})$ to be the *satisfaction relation for atomic formulas*, that is

$$D(\mathcal{A}) = \{\langle i, \bar{a}\rangle : \mathcal{A} \models \varphi_i^{\text{at}}(\bar{a})\} \subseteq \mathbb{HF}_\mathcal{A},$$

where $\{\varphi_0^{\text{at}}, \varphi_1^{\text{at}}, ...\}$ is an effective enumeration of all the atomic τ-formulas. Notice that if the language of $\mathcal{A}$ is finite and relational, this is a finite list of formulas, and hence $D(\mathcal{A})$ is Δ-definable in $\mathcal{HF}_\mathcal{A}$ without using $D(\mathcal{A})$. In that case, there is no need to add D to the language $\mathcal{HF}_\mathcal{A}$, but when τ is infinite, if we do not add D, Σ-formulas can only involve finitely many symbols from τ.

Given any $R \subseteq A^{<\mathbb{N}}$, we can view it directly as a subset of $\mathbb{HF}(\mathcal{A})$. Conversely, there is also a natural way of going from relations in $\mathcal{HF}_\mathcal{A}$ to subsets of $A^{<\mathbb{N}}$. Let $X = \{x_0, x_1, ...\}$, where the x_i's are variable symbols. Every $t \in \mathbb{HF}_X$ is essentially a term over a finite set of variables, and we write $t(\bar{x})$ to show the variables that appear in t. Observe that $\mathbb{HF}_A = \{t(\bar{a}) : t(\bar{x}) \in \mathbb{HF}_X, \bar{a} \in A^{|\bar{x}|}\}$. Let $\{t_i : i \in \mathbb{N}\}$ be an effective enumeration of $\mathbb{HF}_X \cup X$. Now, given $Q \subseteq \mathcal{HF}_\mathcal{A}$, we define

$$s(Q) = \{\langle i, \bar{a}\rangle : t_i(\bar{a}) \in Q\} \subseteq \mathbb{N} \times A^{<\mathbb{N}}.$$

Observation II.4.2. The relation $\{\langle b, \underline{n}, \bar{a}\rangle : b \in \mathbb{HF}_A, n \in \mathbb{N}, \bar{a} \in A^{<\mathbb{N}} \ \&\ b = t_n(\bar{a})\} \subseteq \mathbb{HF}_\mathcal{A} \times \omega \times A^{<\mathbb{N}}$ is Δ-definable in $\mathcal{HF}_\mathcal{A}$. This is not completely trivial, and recursion on terms is necessary.

Theorem II.4.3. *Given $R \in A^{<\mathbb{N}}$, the following are equivalent:*

(1) *R is r.i.c.e. in $\mathcal{A}$.*
(2) *R is Σ-definable in $\mathcal{HF}_\mathcal{A}$ with parameters.*

Given $Q \subseteq A \cup \mathbb{HF}_A$, the following are equivalent:

(1) *$s(Q)$ is r.i.c.e. in $\mathcal{A}$.*
(2) *Q is Σ-definable in $\mathcal{HF}_\mathcal{A}$ with parameters.*

Historical Remark II.4.4. *This theorem is credited to Vaĭtsenavichyus* [**Vaĭ89**] *in* [**Stu**] *and appears in some form in* [**BT79**].

PROOF. We only prove the second part; the proof of the first part is very similar. Suppose first that $s(Q)$ is r.i.c.e. in $\mathcal{A}$. Using Theorem II.1.14, we get a c.e. set W and a tuple $\bar{p} \in A^{<\mathbb{N}}$ such that

$$\langle i, \bar{a} \rangle \in s(Q) \iff \mathcal{A} \models \bigvee_{e:\langle i,e\rangle \in W} \varphi_e^{\exists}(\bar{p}, \bar{a}) \qquad \text{for all } i \in \mathbb{N} \text{ and } \bar{a} \in A^{<\mathbb{N}},$$

where $\{\varphi_e^{\exists} : e \in \mathbb{N}\}$ is an effective enumeration of the $\exists$-τ-formulas. But then $b \in Q$ if and only if $\exists i \in \mathbb{N} \exists \bar{a} \in A^{<\mathbb{N}} \exists e (b = t_i(\bar{a}) \ \& \ \langle i, e\rangle \in W \ \& \ \mathcal{A} \models \varphi_e^{\exists}(\bar{p}, \bar{a}))$. Using that deciding whether $b = t_i(\bar{a})$ is Δ-definable and that W and the existential diagram of $\mathcal{A}$ are Σ-definable, we get that Q is Σ-definable with parameters $\bar{p}$.

Suppose now that Q is Σ-definable in $\mathcal{HF}_{\mathcal{A}}$ with parameters; we want to prove that $s(Q)$ is r.i.c.e.. Let $\mathcal{B}$ be a copy of $\mathcal{A}$. Computably in $D(\mathcal{B})$, build $\mathbb{HF}_{\mathcal{B}}$ and a copy of $\mathcal{HF}_{\mathcal{B}}$, and then use the Σ-definition of Q to enumerate $Q^{\mathcal{HF}_{\mathcal{B}}}$. We end up with a $D(\mathcal{B})$-computable enumeration of $Q^{\mathcal{B}}$, which we can then use to produce a $D(\mathcal{B})$-computable enumeration of $s(Q)$. □

HISTORICAL REMARK II.4.5. *In* [**Mos69**], *Moschovakis introduces what we now call the* Moschovakis enrichment *of a structure $\mathcal{A}$, denoted $\mathcal{A}^*$. For our purposes, there is no real difference between $\mathcal{A}^*$ and $\mathcal{HF}_{\mathcal{A}}$. The difference is that in the iterative definition of the domain of $\mathcal{A}^*$ we take pairs instead of finite subsets as we did for $\mathbb{HF}_{\mathcal{A}}$. Moschovakis* [**Mos69**] *then defines a class of partial multi-valued functions from $(A^*)^n$ to A^* which he calls* search computable functions. *This class is defined as the least class closed under certain primitive operations, much in the style of Kleene's definition of primitive recursive and partial recursive functions, where instead of the Kleene's least-element operator μ, we have a multivalued search operator ν. A subset of A^* is* search computable *if its characteristic function is, and it is* semi-search computable *if it has a definition of the form $\exists y\, (f(x, y) = 1)$, where f is search computable.*

The definition of search computable allows us to add a list of new primitive functions to our starting list (so long as they are given in an effective list, with computable arities), obtaining a sort of relativized version *of search computability. If we have a structure $\mathcal{A}$, we would add to the list of primitive functions the characteristic functions of the relations in $\mathcal{A}$ to obtain a notion of partial, multi-valued,* search computable functions in $\mathcal{A}$.

Much in the same way as we did for $\mathcal{HF}_{\mathcal{A}}$ above, we have a natural way of encoding relations $R \subseteq A^{<\mathbb{N}}$ by subsets of $\mathcal{A}^*$, and vice-versa. Maybe even more directly, one can go from subsets of A^* to subsets of $\mathcal{HF}_{\mathcal{A}}$ and back. Gordon [**Gor70**] *proved that the notions of search computable in $\mathcal{A}$ and semi-search computable in $\mathcal{A}$ for subsets of A^* coincide with the notions of Δ-definable and Σ-definable for subsets of $\mathcal{HF}_{\mathcal{A}}$. Therefore, when we add parameters, they also coincide with the notions of r.i. computable and r.i.c.e. for relations in $A^{<\mathbb{N}}$.*

CHAPTER III

Existentially-atomic models

The key notion in this chapter is that of existentially atomic structures: these are atomic structures where all the types are generated by existential formulas. They are the best-behaved structures around: They are the structures for which it is the easiest to compute isomorphisms between different ω-presentations. They are among the easiest to identify, in the sense that they have simple Scott sentences. Their computational complexity can, in many cases, be simply characterized by a single enumeration degree. But not only they are simple, they are also general: every structure is $\exists$-atomic if one adds enough relations to the language, as for instances if one adds enough jumps, as we will see in Part 2. This means that the results we present in this chapter apply to all structures relative to those relations one needs to add.

In this chapter, we will also introduce a variety of tools that will be useful throughout the book, as for instance the Cantor back-and-forth argument, and the notion of a structure having enumeration degree.

III.1. Definition

Let $\mathcal{A}$ be a τ-structure. The automorphism orbit of a tuple $\bar{a} \in A^{<\mathbb{N}}$ is the set

$$\mathrm{orb}_{\mathcal{A}}(\bar{a}) = \{\bar{b} \in A^{|\bar{a}|} : \text{there is an automorphism of } \mathcal{A} \text{ mapping } \bar{a} \text{ to } \bar{b}\}.$$

Definition III.1.1. A structure $\mathcal{A}$ is $\exists$-*atomic* if, for every tuple $\bar{a} \in A^{<\mathbb{N}}$, there is an $\exists$-formula $\varphi_{\bar{a}}(\bar{x})$ which defines the automorphism orbit of $\bar{a}$; that is,

$$\mathrm{orb}_{\mathcal{A}}(\bar{a}) = \{\bar{b} \in A^{|\bar{a}|} : \mathcal{A} \models \varphi_{\bar{a}}(\bar{b})\}.$$

These structures were studied by Simmons in [**Sim76**, Section 2], and he cites [**Pou72**] as their first occurrence in the literature.

The set of all these defining formulas, $\{\varphi_{\bar{a}} : \bar{a} \in A^{<\mathbb{N}}\}$ makes a Scott family:

Definition III.1.2. A *Scott family* for a structure $\mathcal{A}$ is a set S of formulas such that each $\bar{a} \in A^{<\mathbb{N}}$ satisfies some formula $\varphi(\bar{x}) \in S$, and if $\bar{a}$ and $\bar{b}$ satisfy the same formula $\varphi(\bar{x}) \in S$, they are automorphic.

Thus, a structure is $\exists$-atomic if and only if it has a Soctt family of $\exists$-formulas. Having access to a Scott family for a structure $\mathcal{A}$ allows us to recognize the different tuples in $\mathcal{A}$ up to automorphism. This is exactly what is necessary to build isomorphisms between different copies of $\mathcal{A}$. If we want to build a computable isomorphism, we need the Scott family to be computably enumerable.

Definition III.1.3. We say that a Scott family is *c.e.* if the set of indices for its formulas is c.e. A structure $\mathcal{A}$ is *effectively* $\exists$-*atomic* if it has a c.e. Scott family of $\exists$-formulas.

Example III.1.4. A linear ordering is $\exists$-atomic if and only if it is either finite or dense without end points:

If a linear ordering has n elements, the ith element can be characterized by the $\exists$-formula that says that there $i-1$ elements below it and $n-i-1$ elements above it. If a linear ordering is dense without endpoints, then two tuples are automorphic if and only if they are ordered the same way.

Suppose now that we have a linear ordering that is neither dense nor finite. We claim that there must exist a tuple a, b, c such that: either $a < b < c$, a and b are adjacent, and there are infinitely elements to the right of c; or $c < b < a$, a and b are adjacent, and there are infinitely many elements to the left of c. To prove the claim, we consider three cases: If there is only one adjacency pair in the whole linear ordering, then the linear ordering must have a dense segment; let a and b be the elements of the adjacency pair and take c from the dense segment. If every element has finitely many elements to its right or to its left, then the linear ordering has either an initial segment isomorphic to ω or a final segment isomorphic to ω^*; either let $a < b < c$ be the first three elements, or let $c < b < a$ be the last three. If neither of the above is the case, take c so that it has infinitely many elements to both its left and its right, and let a, b be an adjacency pair disjoint from c. Now that we have proved that a, b, c always exist, we claim that no existential formula defines the orbit of the pair $\langle a, b\rangle$. Notice that any $\exists$-formula true of (a, b) is true of (a, c): To see this, recall the analysis of $\exists$-formulas we did in Lemma II.3.1, and use that the number of elements in each of the intervals $(-\infty, a)$, (a, b), and $(b, +\infty)$ is less than or equal to the number of elements in $(-\infty, a)$, (a, c), and $(c, +\infty)$ respectively. But (a, b) and (a, c) are not automorphic because a and b are adjacent, and a and c are not.

Exercise III.1.5. Prove that the $\exists$-atomic linear orderings with parameters are exactly the finite sums of linear orderings of the form: $\mathbb{N}$, $\mathbb{N}^*$, $\mathbb{Z}$, n and $m \cdot \mathbb{Q}$, for $n, m \in \mathbb{N}$.

Try characterizing which of these are $\exists$-atomic without parameters.

III.2. Existentially algebraic structures

We will see that fields of finite transcendence degree, graphs of finite valance with finitely many connected components, and torsion-free abelian groups of finite rank are all $\exists$-atomic over a finite set of parameters. The reason is that they are $\exists$-algebraic.

Definition III.2.1. An element $a \in A$ is $\exists$-*algebraic* in $\mathcal{A}$ if there is an $\exists$-formula $\varphi(x)$ true of a such that $\{b \in A : \mathcal{A} \models \varphi(b)\}$ is finite. A structure $\mathcal{A}$ is $\exists$-*algebraic* if all its elements are.

Example III.2.2. A *field that is algebraic over its prime sub-field* is $\exists$-algebraic because every element is one of finitely many that is a root of a polynomial over the prime field. We will develop this example further in Example III.8.10.

A *connected graph of finite valance with a selected root vertex* is $\exists$-algebraic because every element is one of finitely many that are at a given distance from the root.

An *abelian torsion-free group with a selected basis* is $\exists$-algebraic because every element is the only one for which a certain non-trivial $\mathbb{Z}$-linear combination of it and the basis evaluates to 0.

We prove that ∃-algebraic structures are ∃-atomic in two lemmas. The core of the arguments is an application of König's lemma that appears in the first one.

LEMMA III.2.3. *Two structures that are ∃-algebraic and have the same ∃-theories are isomorphic.*

PROOF. Let $\mathcal{A}$ and $\mathcal{B}$ be ∃-algebraic structures with the same ∃-theories. To prove that $\mathcal{A}$ and $\mathcal{B}$ are isomorphic, we will define a tree of finite approximations to possible isomorphisms from $\mathcal{A}$ to $\mathcal{B}$, and then use König's lemma to show this tree has a path.

List the elements of A as $\{a_0, a_1, ...\}$. For each n, let $\varphi_n(x_0, ..., x_{n-1})$ be an ∃-formula which is true of tuple $\langle a_0, ..., a_{n-1}\rangle$, has finitely many solutions. (A *solution* to a formula is a tuple that makes it true.) By taking conjunctions if necessary, we may assume that $\varphi_n(x_0, ..., x_{n-1})$ implies $\varphi_{n-1}(x_0, ..., x_{n-2})$. Let

$$T = \{\bar{b} \in B^{<\mathbb{N}} : D_{\mathcal{B}}(\bar{b}) = D_{\mathcal{A}}(a_0, ..., a_{|\bar{b}|-1}) \quad \& \quad \mathcal{B} \models \varphi_{|\bar{b}|}(\bar{b})\}.$$

We will prove that a path through T gives us an isomorphism from $\mathcal{A}$ to $\mathcal{B}$. But before that, let us prove T has a path.

T is clearly a tree in the sense that it is closed under taking initial segments of tuples. It is finitely branching because, for each n, φ_n has finitely many solutions. To show that T is infinite, notice that, for each n,

$$\mathcal{A} \models \exists x_0, ..., x_{n-1}(D(\bar{x}) = \sigma \ \& \ \varphi_n(\bar{x})), \qquad \text{where } \sigma = D_{\mathcal{A}}(a_0, ..., a_{|\bar{b}|-1}),$$

as witnessed by $a_0, ..., a_{n-1}$. Since $\mathcal{A}$ and $\mathcal{B}$ have the same ∃-theories, $\mathcal{B}$ models this sentence too, and the witness is an n-tuple that belongs to T. König's lemma states that every infinite finitely branching tree must have an infinite path. Thus, T must have an infinite path $P \in B^{\mathbb{N}}$. This path determines a map $a_n \mapsto P(n) \colon A \to B$. That map is an embedding because it preserves finite atomic diagrams. But then it must be an isomorphism: If $b \in B$ is a solution of an ∃-formula φ with finitely many solutions, then φ must have the same number of solutions in $\mathcal{A}$ (because $\exists\text{-}Th(\mathcal{A}) = \exists\text{-}Th(\mathcal{B})$), and since ∃-formulas are preserved under embeddings, one of those solutions has to be mapped to b. □

LEMMA III.2.4. *Every ∃-algebraic structure is ∃-atomic.*

PROOF. Let $\mathcal{A}$ be ∃-algebraic and take $\bar{a} \in A^{<\mathbb{N}}$. Let $\varphi(\bar{x})$ be an ∃-formula true of $\bar{a}$ with the least possible number of solutions, say k solutions. We claim that every solution to φ is automorphic to $\bar{a}$. Suppose, toward a contradiction, that $\bar{b}$ satisfies φ but is not automorphic to $\bar{a}$. Then there must be an ∃-formula $\psi(\bar{x})$ that is true of either $\bar{a}$ or $\bar{b}$, but not of both. This is because if $(\mathcal{A}, \bar{a})$ and $(\mathcal{A}, \bar{b})$ satisfied the same ∃-atomic formulas, since they are ∃-algebraic, the previous lemma would imply they are isomorphic. If $\psi(\bar{x})$ is true of $\bar{a}$, then $\varphi(\bar{x}) \wedge \psi(\bar{x})$ would be true of $\bar{a}$ and have fewer solutions than φ, contradicting our choice of φ. If $\psi(\bar{x})$ is not true of $\bar{a}$, and it is true of i out of the k solutions of φ, then the formula of $\bar{x}$ saying

"$\varphi(\bar{x})$ and there are i solutions to $\varphi \wedge \psi$ all different from $\bar{x}$"

is an ∃-formula true of $\bar{a}$ with $k - i$ solutions — getting the desired contradiction. □

The statements of the lemmas in this section are new, but the ideas behind them are not. Proofs like that of Lemma III.2.3 using König's lemma have appeared in many other places before, for instance [**HLZ99**]. The ideas for the proof of Lemma III.2.4 are

similar to those one would use in a proof that algebraic structures are atomic (without the $\exists$-), except that here one has to be slightly more careful.

III.3. Cantor's back-and-forth argument

Before we move on with more on $\exists$-atomic structures, we take an interlude to introduce a tool we will use throughout the book.

DEFINITION III.3.1. Given structures $\mathcal{A}$ and $\mathcal{B}$, we say that a set $I \subseteq \mathcal{A}^{<\mathbb{N}} \times \mathcal{B}^{<\mathbb{N}}$ has the *back-and-forth property* if, for every $\langle \bar{a}, \bar{b} \rangle \in I$,

- $D_{\mathcal{A}}(\bar{a}) = D_{\mathcal{B}}(\bar{b})$ (i.e., $|\bar{a}| = |\bar{b}|$ and $\bar{a}$ and $\bar{b}$ satisfy the same $\tau_{|\bar{a}|}$-atomic formulas);
- for every $c \in A$, there exists $d \in B$ such that $\langle \bar{a}c, \bar{b}d \rangle \in I$; and
- for every $d \in B$, there exists $c \in A$ such that $\langle \bar{a}c, \bar{b}d \rangle \in I$.

(Let us recall that we are using the notation $\bar{a}c$ for the concatenation $\bar{a}^\frown c$.)

The canonical example is the following. If $\mathcal{A}$ and $\mathcal{B}$ are isomorphic, then the set

$$\{(\bar{a}, \bar{b}) \in A^{<\mathbb{N}} \times B^{<\mathbb{N}} : (\mathcal{A}, \bar{a}) \cong (\mathcal{B}, \bar{b})\},$$

has the back-and-forth property. We let the reader verify this fact.

OBSERVATION III.3.2. If follows immediately form the example above, that if $\mathcal{A}$ and $\mathcal{B}$ are isomorphic and S is a Scott family for $\mathcal{A}$, then the set

$$I_{\mathcal{A},\mathcal{B}} = \{(\bar{a}, \bar{b}) \in A^{<\mathbb{N}} \times B^{<\mathbb{N}} : (\text{for some } \varphi \in S)\ \mathcal{A} \models \varphi(\bar{a})\ \&\ \mathcal{B} \models \varphi(\bar{b})\}$$

has the back-and-forth property.

LEMMA III.3.3. *If $I \subseteq \mathcal{A}^{<\mathbb{N}} \times \mathcal{B}^{<\mathbb{N}}$ has the back-and-forth property, then for every $\langle \bar{a}, \bar{b} \rangle \in I$, there is an isomorphism $g \colon \mathcal{A} \to \mathcal{B}$ with $g(\bar{a}) = (\bar{b})$. Moreover, such an isomorphism can be computed from an enumeration of I.*

PROOF. The map $g \colon A \to B$ is defined by stages. Let $\bar{a}_0 = \bar{a}$ and $\bar{b}_0 = \bar{b}$. At each stage $s+1$, we define tuples $\bar{a}_{s+1} \in A^{<\mathbb{N}}$ and $\bar{b}_{s+1} \in B^{<\mathbb{N}}$ with $\bar{a}_s \subseteq \bar{a}_{s+1}$, $\bar{b}_s \subseteq \bar{b}_{s+1}$, and $\langle \bar{a}_{s+1}, \bar{b}_{s+1} \rangle \in I$. The back-and-forth property will allow us to build such sequences in a way that, for every $c \in A$, there is some s such that c is one of the entries of $\bar{a}_s$, and, for every $d \in B$, there is some s such that d is one of the entries of $\bar{b}_s$: All we have to do is take turns choosing elements from $\mathcal{A}$ and $\mathcal{B}$ in such a way that we eventually choose them all. At the end of stages, we define $g \colon A \to B$ so that $g(\bar{a}_s) = \bar{b}_s$. Since $\bar{a}_s$ and $\bar{b}_s$ satisfy the same $\tau_{|\bar{a}_s|}$-atomic formulas, we get that g preserves all the relations, functions, and constants and hence that it is an isomorphism. (Notice that $D_{\mathcal{A}}(\bar{a}_s) = D_{\mathcal{B}}(\bar{b}_s)$ also implies that, if two entries in $\bar{a}_s$ are equal, so are the corresponding ones in $\bar{b}_s$, and hence there is no issue defining g so that it maps $\bar{a}_s$ to $\bar{b}_s$.)

It is clear that g can be computed from an enumeration of I. □

III.4. Uniform computable categoricity

An issue we have to be constantly aware of when working with computable structures is that different copies of same structure may behave differently computationally. Computable categorical structures are the ones where this issue does not show up. They are the ones whose computable copies all have the same computability theoretic properties. We will study them in Chapter VIII. For now, we consider the stronger notion of uniform computable categoricity.

DEFINITION III.4.1. A computable structure $\mathcal{A}$ is *uniformly computably categorical* if there is a computable operator that, when given as an oracle the atomic diagram $D(\mathcal{B})$ of a computable copy $\mathcal{B}$ of $\mathcal{A}$, outputs an isomorphism form $\mathcal{B}$ to $\mathcal{A}$. A computable structure $\mathcal{A}$ is *uniformly relatively computably categorical* if there is a computable operator that, when given $D(\mathcal{B})$ for a (not necessarily computable) copy $\mathcal{B}$ of $\mathcal{A}$, outputs an isomorphism form $\mathcal{B}$ to $\mathcal{A}$.

Notice that if a structure $\mathcal{A}$ has a c.e. Scott family of $\exists$-formulas, and $\mathcal{B}$ is a copy of $\mathcal{A}$, then the set $I_{\mathcal{A},\mathcal{B}}$ from Observation III.3.2 is c.e. in $D(\mathcal{B})$ and has the back-and-forth property. Then, by Lemma III.3.3, we get that $\mathcal{A}$ and $\mathcal{B}$ are $D(\mathcal{B})$-computably isomorphic. Furthermore, the definition of $I_{\mathcal{A},\mathcal{B}}$, and the construction from Lemma III.3.3 are completely uniform, and produce a computable operator as needed in the definition of uniformly relatively computably categoricity.

THEOREM III.4.2 (Ventsov [**Ven92**]). *Let $\mathcal{A}$ be a computable structure. The following are equivalent:*

(1) *$\mathcal{A}$ is effectively $\exists$-atomic.*
(2) *$\mathcal{A}$ is uniformly relatively computably categorical.*
(3) *$\mathcal{A}$ is uniformly computably categorical.*

PROOF. That (1) implies (2) was observed in the previous paragraph. It is obvious that (2) implies (3). The proof that (3) implies (1) is quite a bit more elaborate.

Suppose Γ is a computable operator such that $\Gamma^{D(\mathcal{B})}$ is an isomorphism from $\mathcal{B}$ to $\mathcal{A}$ for every computable copy $\mathcal{B}$ of $\mathcal{A}$. We first claim that for every tuple $\bar{q} \in A^{<\mathbb{N}}$, if $\Gamma^{D_{\mathcal{A}}(\bar{q})}$ converges on $0, ..., k-1$, then

$$\bar{q} \restriction k \text{ is automorphic to } \Gamma^{D_{\mathcal{A}}(\bar{q})} \restriction k,$$

where

$$\Gamma^{D_{\mathcal{A}}(\bar{q})} \restriction k = (\Gamma^{D_{\mathcal{A}}(\bar{q})}(0), \Gamma^{D_{\mathcal{A}}(\bar{q})}(1), ..., \Gamma^{D_{\mathcal{A}}(\bar{q})}(k-1)) \in A^k.$$

Here comes the key observation: the value of $\Gamma^{D_{\mathcal{A}}(\bar{q})} \restriction \bar{n}$ depends only on $D_{\mathcal{A}}(\bar{q}) \in 2^{<\mathbb{N}}$, while it determines the automorphism orbit of $\bar{q} \restriction \bar{n}$. The claim is true because we can always extend $\bar{q}$ to a computable onto map $g \colon \mathbb{N} \to \mathcal{A}$ so that, for $\mathcal{B} = g^{-1}(\mathcal{A})$, $\Gamma^{D(\mathcal{B})}$ is an isomorphism from $\mathcal{B}$ to $\mathcal{A}$. Since g is also an isomorphism from $\mathcal{B}$ to $\mathcal{A}$, the two images of $(0, ..., k-1)$ through those isomorphisms must be automorphic; namely $g \restriction k = \bar{q} \restriction k$ and $\Gamma^{D(\mathcal{B})} \restriction k = \Gamma^{D_{\mathcal{A}}(\bar{q})} \restriction k$ (see figure below).

$$\begin{array}{ccccc} \mathcal{A} & \xleftarrow[\cong]{g} & \mathcal{B} & \xrightarrow[\cong]{\Gamma^{D(\mathcal{B})}} & \mathcal{A} \\ \vdots & & \vdots & & \vdots \\ \bar{q} \restriction k & \longleftarrow\!\shortmid & (0, ..., k-1) & \shortmid\!\longrightarrow & \Gamma^{D_{\mathcal{A}}(\bar{q})} \restriction k \end{array}$$

Now, given a tuple $\bar{a} \in A^{<\mathbb{N}}$, we need to produce an $\exists$-formula defining its orbit, and we need to find this formula computably. Search for $\bar{q} \in A^{<\mathbb{N}}$ extending $\bar{a}$ such that $\Gamma^{D_{\mathcal{A}}(\bar{q})}$ converges on $0, ..., |\bar{a}| - 1$. We now claim that the following formula defines the orbit of $\bar{a}$:

$$\varphi_{\bar{a}}(\bar{x}) \;\equiv\; (\exists \bar{y}) \text{“} D(\bar{x}, \bar{y}) = \sigma, \text{”}$$

where $\sigma = D_{\mathcal{A}}(\bar{q}) \in 2^{<\mathbb{N}}$. (Recall from Observation I.1.8 that, for each $\sigma \in 2^{\ell_{|\bar{z}|}}$, there is a quantifier-free formula $\varphi^{at}_{\sigma}(\bar{z})$ which holds if and only if $D(\bar{z}) = \sigma$.) Clearly, $\bar{a}$ satisfies $\varphi_{\bar{a}}$. Suppose now that $\mathcal{A} \models \varphi_{\bar{a}}(\bar{c})$; we need to show that $\bar{a}$ and $\bar{c}$ are automorphic. Then there is $\bar{p} \supseteq \bar{c}$ such that $D_{\mathcal{A}}(\bar{p}) = \sigma = D_{\mathcal{A}}(\bar{q})$. So

$$\Gamma^{D_{\mathcal{A}}(\bar{p})} \restriction |\bar{a}| = \Gamma^{D_{\mathcal{A}}(\bar{q})} \restriction |\bar{a}|.$$

By our first claim above, the left-hand-side is automorphic to $\bar{p} \restriction |\bar{a}| = \bar{c}$, and the right-hand-side is automorphic to $\bar{q} \restriction |\bar{a}| = \bar{a}$. □

III.5. Existential atomicity in terms of types

The usual definition of atomic models in model theory is in terms of types (as in (A2) below). We show in this section that, for ∃-atomic models, it is enough to look at ∀-types instead of full first-order types.

We need to review some basic definitions. A ∀-*type* on the variables $x_1, ..., x_n$ is a set $p(\bar{x})$ of ∀-formulas with free variables among $x_1, ..., x_n$ that is *consistent*, i.e., that is satisfied by some tuple $a_1, ..., a_n$ in some structure. We say that a ∀-type is *realized in* a structure $\mathcal{A}$ if it is satisfied by some tuple in $\mathcal{A}$. Given $\bar{a} \in A^{<\mathbb{N}}$, the ∀-*type of* $\bar{a}$ *in* $\mathcal{A}$ is the set of ∀-formulas true of $\bar{a}$:

$$\forall\text{-}tp_{\mathcal{A}}(\bar{a}) = \{\varphi(\bar{x}) : \varphi \text{ is a } \forall\text{-formula and } \mathcal{A} \models \varphi(\bar{a})\}.$$

(The obvious assumption here is that $|\bar{x}| = |\bar{a}|$.)

The reason we allow types to be partial is that ∀-types are never complete, as we could not add the negation of ∀-formulas. For the same reason, instead of principal types, we have to deal with supported types.

Definition III.5.1. A type $p(\bar{x})$ is ∃-*supported within a class of structures* $\mathbb{K}$ if there exists an ∃-formula $\varphi(\bar{x})$ which is realized in some structure in $\mathbb{K}$ and which implies all of $p(\bar{x})$ within $\mathbb{K}$; that is, $\mathcal{A} \models \forall\bar{x}(\varphi(\bar{x}) \to \psi(\bar{x}))$ for every $\psi(\bar{x}) \in p(\bar{x})$ and $\mathcal{A} \in \mathbb{K}$. We say that $p(\bar{x})$ is ∃-*supported in a structure* $\mathcal{A}$ if it is ∃-supported in $\mathbb{K} = \{\mathcal{A}\}$.

Theorem III.5.2. *For every structure* $\mathcal{A}$*, the following are equivalent:*

(A1) $\mathcal{A}$ *is* ∃*-atomic.*
(A2) *Every elementary first-order type realized in* $\mathcal{A}$ *is* ∃*-supported in* $\mathcal{A}$*.*
(A3) *Every* ∀*-type realized in* $\mathcal{A}$ *is* ∃*-supported in* $\mathcal{A}$*.*

Proof. It is not hard to see that (A1) implies (A2) and that (A2) implies (A3). Let us prove that (A3) implies (A1).

For each $\bar{a} \in A^{<\mathbb{N}}$, let $\varphi_{\bar{a}}(\bar{x})$ be an ∃-formula supporting the ∀-type of $\bar{a}$. We need to show that $S = \{\varphi_{\bar{a}} : \bar{a} \in A^{<\mathbb{N}}\}$ is a Scott family for $\mathcal{A}$. Notice that $\mathcal{A} \models \varphi_{\bar{a}}(\bar{a})$, because otherwise $\neg\varphi_{\bar{a}}$ would be part of the ∀-type of $\bar{a}$, and hence implied by $\varphi_{\bar{a}}$, which cannot be the case because $\varphi_{\bar{a}}$ is realizable in $\mathcal{A}$. Consider the set

$$I_{\mathcal{A}} = \{(\bar{a}, \bar{b}) \in A^{<\mathbb{N}} \times A^{<\mathbb{N}} : \mathcal{A} \models \varphi_{\bar{a}}(\bar{b})\}.$$

First, let us prove $I_{\mathcal{A}}$ is symmetric; that is, that if $\mathcal{A} \models \varphi_{\bar{a}}(\bar{b})$, then $\mathcal{A} \models \varphi_{\bar{b}}(\bar{a})$. If not, then $\neg\varphi_{\bar{b}}(\bar{x})$ would be part of the ∀-type of $\bar{a}$, and hence implied by $\varphi_{\bar{a}}$. But we know this is not the case because $\bar{b}$ models both $\varphi_{\bar{a}}$ and $\varphi_{\bar{b}}$.

We now claim that $I_{\mathcal{A}}$ has the back-and-forth property (Definition III.3.1). For, suppose $(\bar{a}, \bar{b}) \in I$. Observe $\bar{a}$ and $\bar{b}$ must satisfy the same ∀-types as they both satisfy

$\varphi_{\bar{a}}$ and $\varphi_{\bar{b}}$ which support their respective $\forall$-types. In particular, they satisfy the same $\tau_{|\bar{a}|}$-atomic formulas and hence have the same atomic diagrams. To show the second condition in Definition III.3.1, take $c \in A$. If there was no $d \in B$ with $\langle \bar{a}c, \bar{b}d\rangle \in I_{\mathcal{A}}$, we would have that $\mathcal{B} \models \neg\exists y \varphi_{\bar{a}c}(\bar{b}, y)$. This formula would be part of the $\forall$-type of $\bar{b}$, and hence implied by $\varphi_{\bar{b}}$. But then, since $\mathcal{A} \models \varphi_{\bar{b}}(\bar{a})$, we would have $\mathcal{A} \models \neg\exists y \varphi_{\bar{a}c}(\bar{a}, y)$, which is not true as witnessed by c. The third condition of the back-and-forth property follows from the symmetry of $I_{\mathcal{A}}$.

Finally, to see that S is a Scott family for $\mathcal{A}$, notice that if $\varphi_{\bar{a}}(\bar{b})$ and $\varphi_{\bar{a}}(\bar{c})$ both hold, then, by Lemma III.3.3, both $\bar{b}$ and $\bar{c}$ are automorphic to $\bar{a}$, and hence automorphic to each other. □

III.6. Building structures and omitting types

Before we continue studying the properties of $\exists$-atomic structures, we need to make another stop to prove some general lemmas that will be useful in future sections. First, we prove a lemma that will allow us to find computable structures in a given class of structures. Second, using similar techniques, we prove the type omitting lemma for $\forall$-types, and its effective version.

We need to define one more level of the hierarchy of infinitary formulas.

Definition III.6.1. An *infinitary* Π_2 *formula* (denoted $\Pi_2^{\mathtt{in}}$) is a countable infinite (or finite) conjunction of formulas of the form $\forall \bar{y}\psi(\bar{y}, \bar{x})$, where each formula ψ is $\Sigma_1^{\mathtt{in}}$, and $\bar{x}$ is s fixed tuple of free variables. Such a formula is *computable infinitary* Π_2 (denoted $\Pi_2^{\mathtt{c}}$) if the formulas ψ are $\Sigma_1^{\mathtt{c}}$ and the list of indices of the formulas ψ is computably enumerable. A class of structures is $\Pi_2^{\mathtt{c}}$ if it is the class of all the ω-presentations that satisfy a certain $\Pi_2^{\mathtt{c}}$ sentence.

Assume, without loss of generality, we are working with a relational vocabulary τ. Given a class of structure $\mathbb{K}$, we let $\mathbb{K}^{fin}$ be — essentially — the set of all the finite substructures of the structures in $\mathbb{K}$:

$$\mathbb{K}^{fin} = \{D_{\mathcal{A}}(\bar{a}) : \mathcal{A} \in \mathbb{K}, \bar{a} \in \mathcal{A}^{<\mathbb{N}}\} \subseteq 2^{<\mathbb{N}}.$$

Exercise III.6.2. Show that $\mathbb{K}^{fin}$ is positive-tt equivalent to $\bigcup\{\exists\text{-}Th(\mathcal{A}) : \mathcal{A} \in \mathbb{K}\}$. In particular, they are both Turing and enumeration equivalent. (For the definition of positive-tt reducibility, see page xiii.)

Lemma III.6.3. *Let $\mathbb{K}$ be a $\Pi_2^{\mathtt{c}}$ class for which $\mathbb{K}^{fin}$ is c.e. Then there is at least one computable structure in $\mathbb{K}$.*

Proof. We build a structure in $\mathbb{K}$ by building a finite approximation to it as in Defintion I.1.5. That is, we build a nested sequence of finite structures $\mathcal{A}_s$, $s \in \mathbb{N}$, where $\mathcal{A}_s$ is a τ_{k_s}-structure whose domain is an initial segment of $\mathbb{N}$, and where $k_s \xrightarrow[s\to\infty]{} \infty$. Furthermore, we require that each $\mathcal{A}_s$ be in $\mathbb{K}^{fin}$ (i.e., the diagram of $\mathcal{A}_s$ be in $\mathbb{K}^{fin}$), and that $\mathcal{A}_s \subseteq \mathcal{A}_{s+1}$ (as τ_{k_s}-structures). At the end of stages, we define the τ-structure $\mathcal{A} = \bigcup_{s\in\mathbb{N}} \mathcal{A}_s$.

Let $\bigwedge\!\!\bigwedge_{i\in I} \forall \bar{y}_i \psi_i(\bar{y}_i)$ be the $\Pi_2^{\mathtt{c}}$ sentence that axiomatizes $\mathbb{K}$, where each ψ_i is $\Sigma_1^{\mathtt{c}}$. To get $\mathcal{A} \in \mathbb{K}$, we need to guarantee that, for each each i and each $\bar{a} \in A^{|\bar{y}_i|}$, we have

$\mathcal{A} \models \psi_i(\bar{a})$. For this, when we build $\mathcal{A}_{s+1}$, we will make sure that,

$$(\star) \qquad \text{for every } i < s \text{ and every } \bar{a} \in A_s^{|\bar{y}_i|},\ \mathcal{A}_{s+1} \models \psi_i(\bar{a}).$$

Notice that since ψ_i is $\Sigma_1^{\mathtt{c}}$, $\mathcal{A}_{s+1} \models \psi_i(\bar{a})$ implies $\mathcal{A} \models \psi_i(\bar{a})$. Thus, we would end up with $\mathcal{A} \models \bigwedge\!\!\!\bigwedge_{i\in I} \forall \bar{y}_i \psi_i(\bar{y}_i)$.

Now that we know what we need, let us build the sequence of $\mathcal{A}_s$'s. Suppose we have already built $\mathcal{A}_0, ..., \mathcal{A}_s$ and we want to define $\mathcal{A}_{s+1} \supseteq \mathcal{A}_s$. All we need to do is search for a finite structure in $\mathbb{K}^{fin}$ satisfying $(\star)$, which we can check computably. We need to show that at least one such structure exists. Since $\mathcal{A}_s \in \mathbb{K}^{fin}$, there is some $\mathcal{B} \in \mathbb{K}$ which has a substructure $\mathcal{B}_s$ τ_{k_s}-isomorphic to $\mathcal{A}_s$. Since $\mathcal{B} \models \bigwedge\!\!\!\bigwedge_{i\in I} \forall \bar{y}_i \psi_i(\bar{y}_i)$, for every $i < s$ and every $\bar{b} \in \mathcal{B}_s^{|\bar{y}_i|}$, there exists a tuple in $\mathcal{B}$ witnessing that $\mathcal{B} \models \psi_i(\bar{b})$. Let $\mathcal{B}_{s+1}$ be a finite $\tau_{k_{s+1}}$-substructure of $\mathcal{B}$ containing $\mathcal{B}_s$ and all those witnessing tuples, where $k_{s+1} > k_s$ is so that all the symbols in the $\exists$-disjunct of the ψ_i witnessing $\mathcal{B} \models \psi_i(\bar{b})$ for $i < s$ appear in $\tau_{k_{s+1}}$. Then $\mathcal{B}_{s+1}$ satisfies $(\star)$ with respect to $\mathcal{B}_s$ as needed. □

Corollary III.6.4. *Let $\mathbb{K}$ be a $\Pi_2^{\mathtt{c}}$ class of structures, and S be the $\exists$-theory of some structure in $\mathbb{K}$. If S is c.e. in a set X, then there is an X-computable ω-presentation of a structure in $\mathbb{K}$ with $\exists$-theory S.*

Proof. Add to the $\Pi_2^{\mathtt{c}}$ axiom for $\mathbb{K}$ the $\Pi_2^{\mathtt{c},X}$ sentence saying that the structure must have $\exists$-theory S:

$$\left(\bigwedge\!\!\!\bigwedge_{\text{“}\exists \bar{y}\psi(\bar{y})\text{”}\in S} \exists \bar{y}\psi(\bar{y}) \right) \ \wedge \ \left(\forall \bar{x} \bigvee_{\substack{\sigma\in 2^{\ell_{|\bar{x}|}} \\ \text{“}\exists \bar{y}\varphi_\sigma(\bar{y})\text{”}\in S}} \varphi_\sigma(\bar{x}) \right),$$

where $\varphi_\sigma^{at}(\bar{x})$ is the formula “$D(\bar{x}) = \sigma$” (see Observation I.1.8) and ℓ_s is the length of $D(\bar{a})$ for a tuple $\bar{a}$ of length s. Let $\mathbb{K}_S$ be the new $\Pi_2^{\mathtt{c},X}$ class of structures. All the models in $\mathbb{K}_S$ have $\exists$-theory S, and hence $\mathbb{K}_S^{fin}$ is enumeration reducible to S, and hence it is c.e. in X too. Applying Lemma III.6.3 relative to X, we get an X-computable structure in $\mathbb{K}_S$ as wanted. □

Not only can we build a computable structure in such a class $\mathbb{K}$, we can build one omitting certain types.

Lemma III.6.5. *Let $\mathbb{K}$ be a $\Pi_2^{\mathtt{in}}$ class of structures. Let $\{p_i(\bar{x}_i) : i \in \mathbb{N}\}$ be a sequence of $\forall$-types which are not $\exists$-supported in $\mathbb{K}$. Then there is a structure $\mathcal{A} \in \mathbb{K}$ which omits all the types $p_i(\bar{x}_i)$ for $i \in \mathbb{N}$.*

Furthermore, if $\mathbb{K}$ is $\Pi_2^{\mathtt{c}}$, $\mathbb{K}^{fin}$ is c.e. and the list $\{p_i(\bar{x}_i) : i \in \mathbb{N}\}$ is c.e., we can make $\mathcal{A}$ computable.

Proof. We construct $\mathcal{A}$ by stages as in the proof of Lemma III.6.3, the difference being that now we need to omit the types p_i. So, on the even stages s, we do exactly the same thing we did in Lemma III.6.3, and we use the odd stages to omit the types. That is, we build a sequence of finite τ_{k_s}-structures $\mathcal{A}_0 \subseteq \mathcal{A}_1 \subseteq \cdots$ and at even stages we define $\mathcal{A}_s + 1$ so that it satisfies $(\star)$ from Lemma III.6.3 guaranteeing that $\mathcal{A}$ belongs to $\mathbb{K}$. At odd stages, for $s + 1 = 2\langle i, j\rangle + 1$, we ensure that the jth tuple $\bar{a}$ does not satisfy p_i as follows. For this, we need to define $\mathcal{A}_{s+1}$ so that $\bar{a}$ satisfies some $\exists$-formula

whose negation is in p_i. Let $\bar{b} = \mathcal{A}_s \setminus \bar{a}$, and let $\sigma = D_{\mathcal{A}_s}(\bar{a}, \bar{b})$. So we have that $\bar{a}$ satisfies $\exists \bar{y}\ (D(\bar{a}, \bar{y}) = \sigma)$. Since p_i is not $\exists$-supported in $\mathbb{K}$, there exists a $\forall$-formula $\psi(\bar{x}) \in p_i$ which is not implied by $\exists \bar{y} D(\bar{a}, \bar{y})$ within $\mathbb{K}$. That means that, for some finite $\mathcal{B} \in \mathbb{K}^{fin}$ extending $\mathcal{A}_s$, we have $\mathcal{B} \models \neg\psi(\bar{a})$. Since such $\mathcal{B}$ and ψ exist, we can wait until we find them and then define $\mathcal{A}_{s+1}$ accordingly. □

III.7. Scott sentences of existentially atomic structures.

Existentially atomic structures are also among the simplest ones in terms of the complexity of their Scott sentences.

Definition III.7.1. A sentence ψ is a *Scott sentence* for a structure $\mathcal{A}$ if $\mathcal{A}$ is the only countable structure satisfying ψ.

We will see in Part 2 that every countable structure has a Scott sentence in $\mathcal{L}_{\omega_1,\omega}$. For now, we prove it only for $\exists$-atomic structures.

Lemma III.7.2. *Every $\exists$-atomic structure has a $\Pi^{\mathtt{in}}_2$ Scott sentence. Furthermore, every effectively $\exists$-atomic computable structure has a $\Pi^{\mathtt{c}}_2$ Scott sentence.*

Proof. Let S be a Scott family of $\exists$-formulas for $\mathcal{A}$. For each $\bar{a} \in A^{<\mathbb{N}}$, let $\varphi_{|\bar{a}|}(\bar{x})$ be the $\exists$-formula defining the orbit of $\mathcal{A}$. (For the empty tuple, let $\varphi_\emptyset()$ be a sentence that is always true.) For any other structure $\mathcal{B}$, consider the set

$$I_{\mathcal{B}} = \{(\bar{a}, \bar{b}) \in \mathcal{A}^{<\mathbb{N}} \times \mathcal{B}^{<\mathbb{N}} : \mathcal{B} \models \varphi_{\bar{a}}(\bar{b})\}.$$

If $I_{\mathcal{B}}$ had the back-and-forth property, then, by Lemma III.3.3, we would know that $\mathcal{B}$ is isomorphic to $\mathcal{A}$ (notice that $(\langle\rangle, \langle\rangle) \in I_{\mathcal{B}}$). Recall form the proof of Theorem III.5.2 that $I_{\mathcal{A}}$ has the back-and-forth property. Thus, if $\mathcal{B}$ is isomorphic to $\mathcal{A}$, then so does $I_{\mathcal{B}}$. We get that $I_{\mathcal{B}}$ has the back-and-forth property if and only if $\mathcal{B}$ is isomorphic to $\mathcal{A}$. The Scott sentence for $\mathcal{A}$ says of a structure $\mathcal{B}$ that $I_{\mathcal{B}}$ has the back-and-forth property:

$$\bigwedge_{\bar{a} \in A^{<\mathbb{N}}} \forall x_1, ..., x_{|\bar{a}|} \left(\varphi_{\bar{a}}(\bar{x}) \Rightarrow \left(\forall y \bigvee_{b \in A} \varphi_{\bar{a}b}(\bar{x}y) \right) \wedge \left(\bigwedge_{b \in A} \exists y \varphi_{\bar{a}b}(\bar{x}y) \right) \right).$$

As for the effectivity claim, if $\mathcal{A}$ is a computable ω-presentation and S is c.e., then the map $\bar{a} \mapsto \varphi_{\bar{a}}$ is computable, and the conjunctions and disjunctions in the Scott sentence above are all computable. □

To prove the other direction, we need to go through the type omitting theorem for $\forall$-types.

Theorem III.7.3. *Let $\mathcal{A}$ be a structure. The following are equivalent:*

1. *$\mathcal{A}$ is $\exists$-atomic.*
2. *$\mathcal{A}$ has a $\Pi^{\mathtt{in}}_2$-Scott sentence.*

Proof. We already know that (1) implies (2). For the other direction, suppose ψ is a $\Pi^{\mathtt{in}}_2$ Scott sentence for $\mathcal{A}$, but that $\mathcal{A}$ is not atomic. By Theorem III.5.2, there is a $\forall$-type realized in $\mathcal{A}$ which is not $\exists$-supported. But then, by Lemma III.6.5, there exists a model of ψ which omits that type. This structure could not be isomorphic to $\mathcal{A}$, contradicting that ψ was a Scott sentence for $\mathcal{A}$. □

Lemma III.7.4. *Let $\mathcal{A}$ be a structure. The following are equivalent:*

(1) $\mathcal{A}$ *is* $\exists$*-atomic over a finite tuple of parameters.*
(2) $\mathcal{A}$ *has a* $\Sigma^{\mathtt{in}}_3$*-Scott sentence.*

As the reader might be able to guess by now, a $\Sigma^{\mathtt{in}}_3$*-formula* is a countable disjunction of formulas of the form $\exists \bar{y} \psi(\bar{y}, \bar{x})$, where ψ is $\Pi^{\mathtt{in}}_2$ and $\bar{x}$ is a fixed tuple of variables.

Proof. If $\mathcal{A}$ is $\exists$-atomic over a finite tuple of parameters $\bar{a}$, then $(\mathcal{A}, \bar{a})$ has a $\Pi^{\mathtt{in}}_2$ Scott sentence $\varphi(\bar{c})$. Then $\exists \bar{y} \varphi(\bar{y})$ is a Scott sentence for $\mathcal{A}$.

Suppose now that $\mathcal{A}$ has a Scott sentence $\bigvee_{i \in \mathbb{N}} \exists \bar{y}_i \psi_i(\bar{y}_i)$. $\mathcal{A}$ must satisfy one of the disjuncts, and that disjunct must then also be a Scott sentence for $\mathcal{A}$. So, suppose the Scott sentence for $\mathcal{A}$ is $\exists \bar{y}\ \psi(\bar{y})$, where ψ is $\Pi^{\mathtt{in}}_2$. Let $\bar{c}$ be a new tuple of constants of the same size as $\bar{y}$. If $\varphi(\bar{c})$ were a Scott sentence for $(\mathcal{A}, \bar{a})$, we would know $\mathcal{A}$ is $\exists$-atomic over $\bar{a}$ — but this might not be the case. Suppose $(\mathcal{B}, \bar{b}) \models \varphi(\bar{c})$. Then $\mathcal{B}$ must be isomorphic to $\mathcal{A}$, but we could have $(\mathcal{B}, \bar{b}) \not\cong (\mathcal{A}, \bar{a})$. However, it is enough for us to show that one of the models of $\varphi(\bar{c})$ is $\exists$-atomic over $\bar{c}$. Since there are only countably many models of $\varphi(\bar{c})$, there are countably many $\forall$-types among the models of $\varphi(\bar{c})$. Thus, we can omit the non-$\exists$-supported ones while satisfying $\varphi(\bar{c})$. The resulting structure would be $\exists$-atomic over $\bar{c}$ and isomorphic to $\mathcal{A}$ by Theorem III.5.2. □

III.8. Turing degree and enumeration degree

To measure the computational complexity of a structure, the most common tool is its degree spectrum, which we will study in Chapter V. A much more natural attempt to measure the computational complexity of a structure is given in the following definition — unfortunately, it does not always apply.

Definition III.8.1 (Jockusch and Richter [**Ric81**]). A structure $\mathcal{A}$ has *Turing degree* $X \in 2^{\mathbb{N}}$ if X computes a copy of $\mathcal{A}$, and every copy of $\mathcal{A}$ computes X.

It turns out that if we look at a similar definition, but on the enumeration degrees, we obtain a better behaved notion.

Definition III.8.2. A structure $\mathcal{A}$ has *enumeration degree* $X \subseteq \mathbb{N}$ if every enumeration of X computes a copy of $\mathcal{A}$, and every copy of $\mathcal{A}$ computes an enumeration of X. Recall that an enumeration of X is an onto function $f \colon \mathbb{N} \to X$.

Equivalently, $\mathcal{A}$ has enumeration degree X if and only if, for every Y,

$$Y \text{ computes a copy of } \mathcal{A} \quad \iff \quad X \text{ is c.e. in } Y.$$

Notice that, for $X, Z \subseteq \mathbb{N}$, if $\mathcal{A}$ has enumeration degree X, then $\mathcal{A}$ has enumeration degree Z if and only if X and Z are enumeration equivalent.

Example III.8.3. Given $X \subseteq \mathbb{N}$, the standard example of a structure with enumeration degree X is the *graph* $\mathcal{G}_X$, which is made out of disjoint cycles of different lengths and which contains a cycle of length $n + 3$ if and only if $n \in X$. It is not hard to see that every presentation of this graph can enumerate X: Whenever we find a cycle of length $n + 3$, we enumerate n into X. For the other direction, if we can enumerate X, we can build a copy of $\mathcal{G}_X$ by enumerating a cycle of length $n + 3$ every time we see a number n enter X.

EXAMPLE III.8.4. Given $X \subseteq \mathbb{N}$, consider the group $\mathcal{G}_X = \bigoplus_{i \in X} \mathbb{Z}_{p_i}$, where p_i is the ith prime number. Then $\mathcal{G}_X$ has enumeration degree X: We can easily build $\mathcal{G}_X$ out of an enumeration of X, and for the other direction, we have that $n \in X$ if and only if there exists $g \in \mathcal{G}_X$ of order p_n.

EXERCISE III.8.5. Show that both the graph and the group from the previous examples are $\exists$-atomic.

Note that $\mathcal{A}$ has Turing degree X if and only if has enumeration degree $X \oplus X^c$ (where X^c is the complement of X). This is because $X \leq_T Y \iff X \oplus X^c$ is c.e. in Y. So, in either of the examples above, we can get a graph or a group of Turing degree X by considering $\mathcal{G}_{X \oplus X^c}$. A set X is said to have *total enumeration degree* if it is enumeration equivalent to a set of the form $Z \oplus Z^c$. There are sets which do not have total enumeration degree [**Med55**]. Those are exactly the sets X for which the set $\{Y \in 2^{\mathbb{N}} : X \text{ is c.e. in } Y\}$ has no least Turing degree, as, if Z were the least Turing degree in that set, then X would be enumeration equivalent to $Z \oplus Z^c$ (because we would have that, for all Y, X is c.e. in Y if and only if $Z \leq_T Y$, if and only if $Z \oplus Z^c$ is c.e. in Y). It follows that if a structure has enumeration degree X and X does not have total enumeration degree, then the structure does not have Turing degree.

The enumeration degree of a structure is indeed a good way to measure its computational complexity — if the structure has one. In general, structures need not have enumeration degree. Furthermore, there are whole classes of structures, like linear orderings for instance, where no structure has enumeration degrees unless it is already computable (Section V.1). Before getting into that, the rest of the section is dedicated to classes whose structures all have enumeration degree.

THEOREM III.8.6. *Let $\mathbb{K}$ be a Π^c_2 class, all whose structures are $\exists$-atomic. Then every structure in $\mathbb{K}$ has enumeration degree given by its $\exists$-theory.*

The proof of Theorem III.8.6 needs a couple of lemmas that are interesting on their own right.

LEMMA III.8.7. *Let S be the $\exists$-theory of a structure $\mathcal{A}$. If $\mathcal{A}$ belongs to some Π^c_2 class $\mathbb{K}$ where $\mathcal{A}$ is the only structure with $\exists$-theory S, then $\mathcal{A}$ has enumeration degree S.*

PROOF. By Corollary III.6.4, if X can compute an enumeration of S, then it can compute an ω-presentation of a structure $\mathcal{B} \in \mathbb{K}$ with $\exists$-theory S. By the assumption on $\mathbb{K}$, $\mathcal{A}$ and $\mathcal{B}$ must be isomorphic. So, X is computing a copy of $\mathcal{A}$. Of course, every copy of $\mathcal{A}$ can enumerate S, and hence $\mathcal{A}$ has enumeration degree S. □

LEMMA III.8.8. *If $\mathcal{A}$ and $\mathcal{B}$ are $\exists$-atomic and have the same $\exists$-theory, then they are isomorphic.*

PROOF. We prove that $\mathcal{A}$ and $\mathcal{B}$ are isomorphic using a back-and-forth construction. Let

$$I = \{\langle \bar{a}, \bar{b} \rangle : \forall\text{-}tp_{\mathcal{A}}(a_0, ..., a_s) = \forall\text{-}tp_{\mathcal{B}}(b_0, ..., b_s)\}.$$

We need to show that I has the back-and-forth property (Definition III.3.1). Clearly, $\forall\text{-}tp_{\mathcal{A}}(a_0, ..., a_s) = \forall\text{-}tp_{\mathcal{B}}(b_0, ..., b_s)$ implies $D_{\mathcal{A}}(a_0, ..., a_s) = D_{\mathcal{B}}(b_0, ..., b_s)$. By assumption, $\langle \emptyset, \emptyset \rangle \in I$. For the second condition in Definition III.3.1, suppose $\langle \bar{a}, \bar{b} \rangle \in I$, and let

$c \in A$. Let ψ be the principal $\exists$-formula satisfied by $\bar{a}c$. Since $\forall\text{-}tp_{\mathcal{A}}(\bar{a}) = \forall\text{-}tp_{\mathcal{B}}(\bar{b})$, there is a d in $\mathcal{B}$ satisfying the same formula. We need to show that $\forall\text{-}tp_{\mathcal{A}}(\bar{a}c) = \forall\text{-}tp_{\mathcal{B}}(\bar{b}d)$. Let us remark that since we do not know $\mathcal{A}$ and $\mathcal{B}$ are isomorphic yet, we do not know that ψ generates a $\forall$-type in $\mathcal{B}$.

First, to show $\forall\text{-}tp_{\mathcal{A}}(\bar{a}c) \subseteq \forall\text{-}tp_{\mathcal{B}}(\bar{b}d)$, take $\theta(\bar{x}y) \in \forall\text{-}tp_{\mathcal{A}}(\bar{a}c)$. Then

$$\text{“}\forall y(\psi(\bar{x}y) \to \theta(\bar{x}y))\text{”} \ \in \ \forall\text{-}tp_{\mathcal{A}}(\bar{a}) \ = \ \forall\text{-}tp_{\mathcal{B}}(\bar{b}),$$

and hence $\theta \in \forall\text{-}tp_{\mathcal{B}}(\bar{b}d)$. Let us now prove the other inclusion. Let $\tilde{\psi}(\bar{x}y)$ be the $\exists$-formula generating $\forall\text{-}tp_{\mathcal{B}}(\bar{b}d)$. Then since $\neg\tilde{\psi} \notin \forall\text{-}tp_{\mathcal{B}}(\bar{b}d)$, by our previous argument, $\neg\tilde{\psi} \notin \forall\text{-}tp_{\mathcal{A}}(\bar{a}c)$ either, and hence $\mathcal{A} \models \tilde{\psi}(\bar{a}c)$. The rest of the proof that $\forall\text{-}tp_{\mathcal{B}}(\bar{b}d) \subseteq \forall\text{-}tp_{\mathcal{A}}(\bar{a}c)$ is now symmetrical to the one of the other inclusion: For $\tilde{\theta}(\bar{x}y) \in \forall\text{-}tp_{\mathcal{A}}(\bar{b}d)$, we have that $\text{“}\forall y(\tilde{\psi}(\bar{x}y) \to \tilde{\theta}(\bar{x}y))\text{”} \in \forall\text{-}tp_{\mathcal{A}}(\bar{a}c)$, and hence $\theta \in \forall\text{-}tp_{\mathcal{B}}(\bar{a}c)$. □

Proof of Theorem III.8.6. The proof is immediate from Lemmas III.8.7 and III.8.8. □

The following gives a structural property that is sufficient for a structure to have enumeration degree. The property is far from necessary though.

Exercise III.8.9. Suppose that a structure $\mathcal{A}$ has a Σ^{c}_3 Scott sentence. prove that $\mathcal{A}$ has enumeration degree.

Example III.8.10 (Frolov, Kalimullin and R. Miller [**FKM09**]). Consider the class $\mathbb{K}$ of fields of finite transcendence degree over $\mathbb{Q}$. This class is not Π^{c}_2, but if we consider $\mathbb{K}_n$ to be the class of fields of transcendence degree n, and add n constant symbols to name a transcendence basis, $v_1, ..., v_n$, then we do get a Π^{c}_2 class. Since all these fields are algebraic over $\mathbb{Q}(v_1, ..., v_n)$, they are $\exists$-algebraic, and hence $\exists$-atomic. It then follows from Theorem III.8.6 that every such field has enumeration degree, namely the enumeration degree of the $\exists$-type of a transcendence basis.

Furthermore, for every set X, there is an algebraic field whose $\exists$-theory is enumeration-equivalent to X: Take the field that contains the p_nth roots of unity if and only if $n \in X$, where p_n is the nth prime number. Clearly, from an enumeration of X, one can build such a field, and hence enumerate its $\exists$-theory, and conversely, the $\exists$-theory of that field can enumerate X.

Example III.8.11 (Calvert, Harizanov, Shlapentokh [**CHS07**]). Torsion-free abelian groups of finite rank always have enumeration degree. If we add a base of the group as parameters, then the class of torsion-free abelian groups generated by such a base is Π^{c}_2. These groups are clearly $\exists$-algebraic and $\exists$-atomic, as every element is generated as a $\mathbb{Q}$-linear combination of the base. Thus, they have enumeration degree.

Furthermore, for every set X there is a torsion-free abelian group of rank one with enumeration degree X: Consider the subgroup of $\mathbb{Q}$ generated by $1/p_n$ for $n \in X$.

Example III.8.12 (Steiner [**Ste13**]). Graphs of finite valance with finitely many connected components always have enumeration degree and can have all possible enumeration degrees: First, we need to add a constant element for each connected component. Now, saying that every element is connected to one of these elements becomes Π^{c}_2. However, saying that the group is finite valance is not Π^{c}_2. But the $\forall$-theory of a graph of finite valance says that it has finite valance: for each constant element, and

for each $k \in \mathbb{N}$, is says that there is a certain finite number of nodes at distance k from that constant. For the same reason, these graphs $\exists$-algebraic, and hence $\exists$-atomic. It then follows from Lemma III.8.7 that they have enumeration degree.

One can show that for every X there is a connected graph of finite valance and enumeration degree X. The graphs $\mathcal{G}_X$ from III.8.3 are not connected, but a small modification would work. We let the reader figure this one out.

Exercise III.8.13. Show that if $\mathcal{A}$ is $\exists$-atomic and has enumeration degree, then the enumeration degree is given by it $\exists$-theory. Hint in footnote.[1]

[1]Show that every $\exists$-type is e-reducible to the $\exists$-theory of $\mathcal{A}$.

CHAPTER IV

Generic presentations

Forcing and generics are a useful tool all over computability theory. The first forcing-style argument in computability theory can be traced back to the Kleene–Post construction [**KP54**] of two incomparable degrees — published a decade before the invention of forcing. In this chapter, we give an introduction to forcing in computable structure theory. We will develop a more general framework for forcing in Part 2, once we gain more familiarity with infinitary languages. For now, instead of looking at fully generic objects, we consider 1-generics, which have relatively low computational complexity.

The notion of forcing was introduced by Cohen to prove that the continuum hypothesis does not follow from the axioms of set theory in ZFC. Soon after, forcing became one of the main tools in set theory to prove independence results of all kinds. Generic objects are "generic" or "typical" in the sense that they do not have any property that is satisfied by a meager class of objects, where meagerness is viewed as a notion of smallness. This implies that if a generic satisfies a particular property, there is a clear reason that forces it to have the property, and it is never by coincidence. Our forcing arguments will essentially have that form: if a generic presentation has a certain computational property, then there must be a synthactical reason for it.

Generic objects come in all different shapes and sizes, but here, we will only consider Cohen generics. A *Cohen generic real* is a real in $\mathbb{N}^{\mathbb{N}}$ that does not belong to any meager set, where a subset of $\mathbb{N}^{\mathbb{N}}$ is *meager* if it is contained in a countable union of nowhere-dense closed sets, and a set is *nowhere dense* if it is not dense when restricted to any open set. Meager sets are considered to be *small sets* — for instance, Baire's category theorem states that no countable union of meager set can cover all of $\mathbb{N}^{\mathbb{N}}$. If a real belongs to a particular meager set, this would a particular property this real has that most reals do not have. The key property that generics have is the following: If $G \in \mathbb{N}^{\mathbb{N}}$ is generic, $P \subseteq \mathbb{N}^{\mathbb{N}}$ is a property, and $G \in P$, then there is a finite initial segment $\sigma \subseteq G$ which *forces* G to belong to P in the sense that every generic extending σ belongs to P. One problem that arises is that every real belongs to a meager set, namely the singleton that contains itself. That is why in set theory one has to work with generic reals that live outside the universe of sets. For the purposes of computability theory, we do not need to consider all meager sets, but only countably many.

We start this chapter by reviewing *1-generic reals*; these are the ones that avoid all nowhere-dense closed sets given as the boundaries of effectively open sets (Definition IV.1.1). They were introduced by Jockusch [**Joc80**], but the construction of Kleene–Post [**KP54**] already gives 1-generic reals 26 years earlier. See Exercise IV.1.7 below for a proof of Kleene–Post's result that every countable partial ordering embeds into the Turing degrees using 1-generics. They where then used in all kinds of embeddability

results into the Turing degrees and other kind of degrees. They are also often used in effective randomness and in reverse mathematics.

The objective of this chapter, though, is to introduce *1-generic enumerations* and *1-generic presentations* of structures, which are similar to the notions originally considered independently by Knight [**Kni86**], and by Manasse and Slaman (later published in [**AKMS89**]). We will develop a more general notion of forcing and generics later in Part 2. For now, 1-generic presentation are enough for the results in this first part of the book. We will use them in the next chapter to prove Richter's theorem V.1.7, Knight et al.'s theorem V.3.1 and Andrews and Miller's theorem V.3.6.

IV.1. Cohen Generic reals

W review the standard notion of 1-genericity on reals and prove some of their basic properties. We will extend these proofs to generic enumerations of structures in the next sections.

For $R \subseteq \mathbb{N}^{<\mathbb{N}}$, define the *open subset of* $\mathbb{N}^{\mathbb{N}}$ *generated by* R to be

$$[R] = \{X \in \mathbb{N}^{\mathbb{N}} : \exists \sigma \in R \ (\sigma \subset X)\}.$$

A subset of $\mathbb{N}^{\mathbb{N}}$ is *effectively open* if it is of the form $[R]$ for some c.e. $R \subseteq \mathbb{N}^{<\mathbb{N}}$. A real $G \in \mathbb{N}^{\mathbb{N}}$ is *1-generic* if and only if it avoids the boundaries of all effectively open sets. Thus, for every effectively open set, either G is well inside it or well outside it. Here is an equivalent, more combinatorial definition.

Definition IV.1.1 (Jockusch [**Joc80**]). We say that a string $\gamma \in \mathbb{N}^{<\mathbb{N}}$ *decides* a subset $R \subseteq \mathbb{N}^{<\mathbb{N}}$ if either there exists $\sigma \subseteq \gamma$ with $\sigma \in R$ or, for all $\sigma \supseteq \gamma$, $\sigma \notin R$. A real $G \in \mathbb{N}^{\mathbb{N}}$ is *1-generic* if for every c.e. subset R of $\mathbb{N}^{<\mathbb{N}}$, there is an initial string of G, $G \upharpoonright k$ for some k, which decides R.

The reason we use the words "decide" and "force" is the following: Let G be 1-generic and $[R]$ be an effectively open set. For $\gamma \subset G$, if $(\exists \sigma \subseteq \gamma)\ \sigma \in R$, we say that γ *forces* G *to be in* $[R]$, while if $(\forall \sigma \nmid \gamma)\ \sigma \notin R$, then we say that γ *forces* G *to be outside* $[R]$. (Recall that $\sigma \nmid \gamma$ means that σ and γ are compatible, i.e., that either $\sigma \subseteq \gamma$ or $\gamma \subseteq \sigma$.) In either case, γ *decides* whether G belongs to $[R]$ or not.

One can require more genericity by requiring G to decide more sets, e.g., α-generics decide all Σ^0_α sets R, as we will see in Part 2. *Cohen generics* decide all sets R in the universe — we will not deal with these in this book.

Observation IV.1.2. 1-generic reals are not computable: For each computable $C \in \mathbb{N}^{\mathbb{N}}$, consider $R_C = \{\sigma \in \mathbb{N}^{<\mathbb{N}} : \sigma \not\subset C\}$. Since there is not enough room in R_C to force out of it, any 1-generic must be be forced to be in $[R_C]$ and hence different from C.

Lemma IV.1.3. *There is a 1-generic real computable from* $0'$.

Proof. This is essentially an effective version of the Baire category theorem.

We build a 1-generic G as the union of a nested sequence of finite strings $\bar{p}_0 \subseteq \bar{p}_1 \subseteq \cdots \in \mathbb{N}^{<\mathbb{N}}$. Let $\bar{p}_0$ be the empty string. At stage $s+1 = e$, we define $\bar{p}_{s+1}$ so that it decides the eth c.e. set $W_e \subseteq \mathbb{N}^{<\mathbb{N}}$: If there is a $\bar{q} \supseteq \bar{p}_s$ with $\bar{q} \in W_e$, we let $\bar{p}_{s+1} = \bar{q}$. Otherwise, we let $\bar{p}_{s+1} = \bar{p}_s$. At the end of stages, we define $G = \bigcup_s \bar{p}_s$. It is not hard

to check that G is 1-generic. (To see that the lengths of the $\bar{p}_s$'s go to infinity, notice that the set $\{\sigma \in \mathbb{N}^{<\mathbb{N}} : |\sigma| \geq n\}$ is c.e. for every n, and hence is eventually considered as one of the W_e's.)

The only step in the construction that was not computable was checking whether there existed $\bar{q} \supseteq \bar{p}_s$ with $\bar{q} \in W_e$. This is a question $0'$ can answer, and hence the whole construction is computable in $0'$. □

For the next lemma, we need to consider the relativized version of 1-genericity. Given $X \in \mathbb{N}^{\mathbb{N}}$, we say that $G \in \mathbb{N}^{\mathbb{N}}$ is *X-1-generic* if every X-c.e. subset of $\mathbb{N}^{<\mathbb{N}}$ is decided by an initial segment of G. The next lemma says that generics do not enumerate new c.e. sets.

LEMMA IV.1.4. *Let $G, X \in \mathbb{N}^{\mathbb{N}}$. Suppose that G is X-1-generic. Then X is not c.e. in G, unless X is c.e. already.*

PROOF. Suppose that $X = W_e^G$ for some $e \in \mathbb{N}$; we will show that X is already c.e. Consider the set of strings which "force '$W_e^G \not\subseteq X$.' "

$$Q = \{\bar{q} \in \mathbb{N}^{<\mathbb{N}} : \exists n \ (n \in W_e^{\bar{q}} \wedge n \notin X)\}.$$

Notice that Q is c.e. in X, and hence it is decided by some initial segment of G — say by $G \upharpoonright k$. If we had $G \upharpoonright k \in Q$, we would get $n \in W_e^G$ and $n \notin X$, contradicting our assumption. Thus, no extension of $G \upharpoonright k$ is in Q.

We now claim that

$$X = \{n \in \mathbb{N} : (\exists \bar{q} \supseteq G \upharpoonright k) \ n \in W_e^{\bar{q}}\}.$$

Notice that this would show that X is c.e. as needed. As for the claim: If $n \in X$, then, since $X = W_e^G$, there is some initial segment $\bar{q}$ of G satisfying $n \in W_e^{\bar{q}}$. For the other inclusion, if there exists $\bar{q} \supseteq G \upharpoonright k$ with $n \in W_e^{\bar{q}}$, then n must belong to X as otherwise $\bar{q}$ would be an extension of $G \upharpoonright k$ in Q. □

In particular, we get that if G is X-1-generic, then G computes X if and only if X is computable (using that computable is equivalent to c.e. and co-c.e.). Thus, if G is X-1-generic, G and X form a minimal pair (i.e., there is no non-computable set computable from both): This is because if $Y \leq_T X$, then G is Y-1-generic too.

The following lemma shows that 1-generics do not code much information on their jumps.

LEMMA IV.1.5. *Every 1-generic real G is* generalized low*; that is, $G' \equiv_T G \oplus 0'$.*

PROOF. That $G' \geq_T G \oplus 0'$ is true for all reals G. Let us prove that $G' \leq_T G \oplus 0'$. Take $e \in \mathbb{N}$; we want to decide if $\Phi_e^G(e) \downarrow$ using $G \oplus 0'$ as oracle uniformly in e. Consider the set

$$R_e = \{\bar{q} \in \mathbb{N}^{<\mathbb{N}} : \Phi_e^{\bar{q}}(e) \downarrow\}.$$

Since R_e is c.e., it is decided by G. Notice that $0'$ can tell if a string γ decides R_e or not, and if it does, whether it forces $G \in [R]$ or $G \notin [R]$. Then, using $G \oplus 0'$, we can find $k \in \mathbb{N}$ such that $G \upharpoonright k$ decides R_e. If $G \upharpoonright k \in R_e$, we know that $\Phi_e^G(e) \downarrow$ and hence $e \in G'$. If no extension of $G \upharpoonright k$ is in R_e, then $\Phi_e^G(e) \uparrow$ and hence $e \notin G'$. □

The following lemma shows that if we split a 1-generic in two pieces, then not only the pieces are 1-generic themselves, but also 1-generic relative to each other.

LEMMA IV.1.6. *Let $G, H \in \mathbb{N}^{\mathbb{N}}$. Then $G \oplus H$ is 1-generic if and only if G is 1-generic and H is G-1-generic.*

PROOF. Suppose first that $G \oplus H$ is 1-generic. Consider a c.e. operator W which outputs subsets of $\mathbb{N}^{<\mathbb{N}}$. To prove that H is G-1-generic, we need to show that H decides W^G using the genericity of $G \oplus H$. Consider the c.e. set of pairs of string that force $H \in [W^G]$:

$$R = \{\gamma \oplus \delta \in \mathbb{N}^{<\mathbb{N}} : \delta \in W^\gamma\}.$$

$G \oplus H$ must decide R. If we have $\gamma \oplus \delta \subset G \oplus H$ with $\gamma \oplus \delta \in [R]$, then $\delta \in W^G$ and H is forced into $[W^G]$. If we have that $(\forall \tau \supseteq \gamma \oplus \delta)\ \tau \notin R$, then $(\forall \sigma \supseteq \delta)\ \sigma \notin W^G$ and H is forced out of $[W^G]$.

In exactly the same way we can show that G is H-1-generic, and in particular 1-generic.

For the other direction, suppose G is 1-generic and H is G-1-generic. Let R be a c.e. subset of $\mathbb{N}^{<\mathbb{N}}$; we must prove that $G \oplus H$ decides it. Assume R is closed upward under inclusion of strings; this is without loss of generality as deciding R is equivalent to deciding its upward closure. Define

$$S_1 = \{\delta \in \mathbb{N}^{<\mathbb{N}} : (G \upharpoonright |\delta|) \oplus \delta \in R\}.$$

S_1 is c.e. in G and thus H must decide it. If there is a $\delta_1 \subset H$ with $\delta \in S_1$, then $G \upharpoonright |\delta_1| \oplus \delta_1$ forces $G \oplus H$ to be in $[R]$. So, suppose there is $\delta_1 \subset H$ no extension of which is in S_1. Define

$$S_0 = \{\gamma \in \mathbb{N}^{<\mathbb{N}} : \exists \delta \in \mathbb{N}^{<\mathbb{N}}\ (\delta \supseteq \delta_1\ \&\ |\delta| = |\gamma|\ \&\ \gamma \oplus \delta \in R\}.$$

G must decide S_0. There cannot be a $\gamma \subseteq G$ with $\gamma \in S_0$, because the witness δ would be an extension of δ_1 in S_1. So, there is $\gamma \subseteq G$ no extension of which is in S_0, and hence we get that $\gamma \oplus (H \upharpoonright |\gamma|)$ forces $G \oplus H$ out of $[R]$. □

Such H and G are said to be *mutually generic*. Similarly, we can get an infinite sequence of mutually generic reals by taking the columns $\{G^{[n]} : n \in \mathbb{N}\}$ of a 1-generic G.

EXERCISE IV.1.7. Prove Kleene–Post's theorem that every countable partial ordering embeds into the Turing degrees. To prove it, given a partial ordering $(P, \leq_P)$, consider a bijection $f\colon P \times \mathbb{N} \to \mathbb{N}$, and consider the pull-back $H = f^{-1}(G)$ of a 1-generic real $G \subseteq \mathbb{N}$. Show that the map $p \mapsto \bigoplus_{q \leq_P p} H^{[q]}\colon P \to \mathbb{N}^{\mathbb{N}}$ induces the desired embedding.

EXERCISE IV.1.8. Prove that the countable atomless Boolean algebra embeds into the Turing degrees preserving joins and meets.Hint in footnote.[1]

IV.2. Generic enumerations

We now move to consider generic enumerations of structures. The main difference with 1-generics reals, is that instead of deciding the c.e. subsets of $\mathbb{N}^{<\mathbb{N}}$, we now decide the r.i.c.e. subsets of $A^{<\mathbb{N}}$.

[1]Consider a 1-generic subset H of $\mathbb{Q}$ and then given an element a of the interval algebra of $\mathbb{Q}$, map it to $a \cap H$.

We assume throughout the rest of the chapter that $\mathcal{A}$ is an ω-presentation of a τ-structure. Given a set A, let $A^\star$ be the set of all finite strings from A whose entries are all different:

$$A^\star = \{\sigma \in A^{<\mathbb{N}} : (\forall i \neq j < |\sigma|)\ \sigma(i) \neq \sigma(j)\}.$$

DEFINITION IV.2.1. We say that $\gamma \in A^\star$ *decides* a subset $R \subseteq A^\star$ if either there is $\sigma \subseteq \gamma$ with $\sigma \in R$ or, for all $\sigma \supseteq \gamma$, we have $\sigma \notin R$. We say that a one-to-one function $g \in A^{\mathbb{N}}$ is a *1-generic enumeration of* $\mathcal{A}$ if, for every r.i.c.e. set $R \subseteq A^\star$, there is an initial segment of g which decides R.

The existence of 1-generic enumerations follows from the Baire category theorem. As in Lemma IV.1.3, we can build a 1-generic enumeration of $\mathcal{A}$ computably in $D(\mathcal{A})'$ by finite approximations deciding all $D(\mathcal{A})$-c.e. sets. Since we only need to decide the r.i.c.e. sets, we can do this with less than $D(\mathcal{A})'$: The lemma below says that $\vec{K}^{\mathcal{A}}$ is enough. See II.2.3 for the definition of the complete r.i.c.e. set $\vec{K}^{\mathcal{A}}$ from Definition, and recall that we always have $\vec{K}^{\mathcal{A}} \leq_T D(\mathcal{A})'$, but that sometimes we have $\vec{K}^{\mathcal{A}} <_T D(\mathcal{A})'$ (Exercise II.2.6).

LEMMA IV.2.2. *Every ω-presentation $\mathcal{A}$ has a 1-generic enumeration computable in $\vec{K}^{\mathcal{A}}$.*

PROOF. We build g as the union of a strictly increasing sequence $\{\bar{p}_s : s \in \mathbb{N}\}$ with $\bar{p}_s \in A^\star$. Recall from Remark II.2.4 that there is a $D(\mathcal{A})$-effective enumeration $\{R_0, R_1, ...\}$ of the r.i.c.e. subsets of $A^\star$. At stage $s+1 = e$, we define $\bar{p}_{s+1}$ to decide the eth r.i.c.e. set $R_e \subseteq A^\star$ as follows: If there is a $\bar{q} \supseteq \bar{p}_s$ with $\bar{q} \in R_e$, we let $\bar{p}_{s+1} = \bar{q}$. Otherwise, we let $\bar{p}_{s+1} = \bar{p}_s$. Finally, we let $g = \bigcup_s \bar{p}_s \in A^{\mathbb{N}}$. It is not hard to check that g is one-to-one and 1-generic.

To carry on this construction, we need to check at each stage $s+1$ whether there exists $\bar{q} \supseteq \bar{p}_s$ with $\bar{q} \in R_e$ or not. The set of $\bar{p}$'s such that $\exists \bar{q} \supseteq \bar{p}(\bar{q} \in R_e)$ is Σ^{c}_1-definable and its index can be obtained uniformly from e. Hence, $\vec{K}^{\mathcal{A}}$ can decide whether $\bar{p}_s$ belongs to it or not, and thus, the whole construction is computable in $\vec{K}^{\mathcal{A}}$. □

It is not hard to see that a 1-generic enumeration must be onto (the set $\{\bar{p} \in A^\star : \bigvee_{i<|\bar{p}|} \bar{p}(i) = a\}$ is r.i.c.e. for all $a \in A$), and hence that it indeed is an enumeration of A. Using the pull-back (see Section I.1.7), each 1-generic enumeration induces what we call a 1-generic presentation:

DEFINITION IV.2.3. An ω-presentation $\mathcal{C}$ is a *1-generic presentation* of $\mathcal{A}$ if it is the pull-back $g^{-1}(\mathcal{A})$ of some 1-generic enumeration g of $\mathcal{A}$.

The reason we defined 1-generic enumerations of $\mathcal{A}$ using r.i.c.e. sets, instead of $D(\mathcal{A})$-c.e. sets, is that we get a notion that is independent of the given ω-presentation of $\mathcal{A}$:

LEMMA IV.2.4. *Let $\mathcal{A}$ and $\mathcal{B}$ be isomorphic. Any 1-generic presentation of $\mathcal{A}$ is also a 1-generic presentation of $\mathcal{B}$.*

PROOF. Let $h\colon \mathcal{A} \to \mathcal{B}$ be an isomorphism. The key point is that h preserves Σ^{c}_1-definable sets.

Suppose that $g\colon \mathbb{N} \to A$ is a 1-generic enumeration of $\mathcal{A}$, and let $\mathcal{C} = g^{-1}(\mathcal{A})$. We want to show that $\mathcal{C}$ is a 1-generic presentation of $\mathcal{B}$ too. Since $\mathcal{C} = (h \circ g)^{-1}(\mathcal{B})$, it is enough to show that $h \circ g$ is a 1-generic enumeration of $\mathcal{B}$. Let $R \subseteq B^\star$ be Σ^{c}_1-definable in $\mathcal{B}$ with parameters; we need to show that $h \circ g$ decides it. Since h is an isomorphism, $h^{-1}(R) \subseteq A^\star$ is Σ^{c}_1-definable in $\mathcal{A}$ with parameters, and hence decided by g. Let $k \in \mathbb{N}$ be such that either $g \upharpoonright k \in h^{-1}(R)$ or, for all $\sigma \in A^\star$ with $\sigma \supseteq g \upharpoonright k$, we have $\sigma \notin h^{-1}(R)$. Applying h, we get that $(h \circ g) \upharpoonright k$ decides R, as wanted. □

In particular, a 1-generic presentation of a structure $\mathcal{A}$ is also a 1-generic presentation of itself. Thus, an ω-presentation $\mathcal{C}$ is a 1-generic presentation if and only if every r.i.c.e. set $R \subseteq C^\star = \mathbb{N}^\star$ is decided by some tuple of the form $\langle 0, 1, ..., k-1\rangle$.

IV.3. Relations on generic presentations

Generic presentations are useful because whatever happens to them, happens for a reason. For instance, we will see that if a relation is c.e. on a generic presentation, it is because it was r.i.c.e. already (assuming the ω-presentation is generic relative to the relation too). In Theorem II.1.14, we showed that a relation $R \subseteq A^{<\mathbb{N}}$ is Σ^{c}_1-definable with parameters if and only if $R^{\mathcal{B}}$ is c.e. in $D(\mathcal{B})$ for every $(\mathcal{B}, R^{\mathcal{B}}) \cong (\mathcal{A}, R)$ (i.e., it is r.i.c.e.). The following theorem, which is the analog of Lemma IV.1.4, shows that we do not need to consider all the copies of $(\mathcal{A}, R)$, but just one that is 1-generic. The proof of Ash–Knight–Manasse-Slaman; Chisholm's Theorem II.1.14 can then be though of as building a 1-generic copy of $(\mathcal{A}, R)$.

Theorem IV.3.1. *Let $\mathcal{A}$ be a structure and $R \subseteq A^{<\mathbb{N}}$. Suppose $(\mathcal{A}, R)$ is a 1-generic presentation. Then R is c.e. in $D(\mathcal{A})$ if and only if R is r.i.c.e.*

Proof. Clearly, if R is r.i.c.e. it is c.e. in $D(\mathcal{A})$. Let us prove the other direction.

Suppose that $R = W_e^{D(\mathcal{A})}$ for some $e \in \mathbb{N}$. Consider the same set we used in the proof of Theorem II.1.14, in which we were trying to build a generic enumeration $\mathcal{C}$ of $\mathcal{A}$ satisfying $W_e^{D(\mathcal{C})} \not\subseteq R^{\mathcal{C}}$:

$$Q = \{\bar{q} \in A^\star : \exists \ell, j_1, ..., j_\ell < |\bar{q}| \left(\langle j_1, ..., j_\ell\rangle \in W_e^{D_{\mathcal{A}}(\bar{q})} \text{ and } \langle q_{j_1}, ..., q_{j_\ell}\rangle \notin R\right)\}.$$

It is not hard to see that Q is r.i.c.e. in $(\mathcal{A}, R)$. So Q is decided by some tuple of the form $\langle 0, ..., k-1\rangle \in A^\star$. We cannot have $\langle 0, ..., k-1\rangle \in Q$, as otherwise there would be a tuple $\langle j_1, ..., j_\ell\rangle \in W_e^{D(\mathcal{A})}$ with $\langle j_1, ..., j_\ell\rangle \notin R$, contradicting that $R = W_e^{D(\mathcal{A})}$. Thus, no extension of $\langle 0, ..., k-1\rangle$ is in Q. Let $\bar{p} = \langle 0, ..., k-1\rangle$. It now follows from Claim 1 inside the proof of Theorem II.1.14 that R is Σ^{c}_1-definable in $\mathcal{A}$ with parameters $\bar{p}$ as needed. To be more explicit, recall that the proof of Claim 1 went through proving that

$$R = \{(q_{j_1}, ..., q_{j_\ell}) : \text{ for } \bar{q} \in A^\star \text{ and } \ell, j_1, ..., j_\ell < |\bar{q}|, \text{ with } \bar{q} \supseteq \bar{p} \text{ and } (j_1, ..., j_\ell) \in W_e^{D_{\mathcal{A}}(\bar{q})}\}. \quad \square$$

Recall that a set $X \subseteq \mathbb{N}$ is coded by $\mathcal{A}$ if and only if it is c.e. in every presentation of $\mathcal{A}$ (see Subsection II.1.4). This is equivalent to saying that $\vec{X}$ is r.i.c.e. in $\mathcal{A}$, where $\vec{X}$ is the subset of $A^{<\mathbb{N}}$ capturing X (Definition II.1.17).

Corollary IV.3.2. *Let $X \subseteq \mathbb{N}$ and suppose $(\mathcal{A}, \vec{X})$ is a 1-generic presentation. Then X is c.e. in $D(\mathcal{A})$ if and only if it is coded by $\mathcal{A}$.*

PROOF. Immediate from the previous theorem. □

Let us remark that saying that $(\mathcal{A}, \vec{X})$ is a 1-generic presentation is equivalent to saying that $\mathcal{A}$ is X-1-generic.

COROLLARY IV.3.3. *For every ω-presentation $\mathcal{B}$, there is another ω-presentation $\mathcal{A} \cong \mathcal{B}$ such that, a set $X \subseteq \mathbb{N}$ is c.e. in both $D(\mathcal{A})$ and $D(\mathcal{B})$ if and only if it is coded by $\mathcal{B}$.*

PROOF. If X is coded by $\mathcal{B}$, by definition it is c.e. in $D(\mathcal{B})$ and in $D(\mathcal{A})$ for every copy $\mathcal{A}$ of $\mathcal{B}$.

For the other direction, let $Y = D(\mathcal{B})'$ and let $\mathcal{A}$ be Y-1-generic. If X is c.e. in $D(\mathcal{B})$, then $X \leq_T Y$ and hence $\mathcal{A}$ is 1-X-generic. If X is also c.e. in $D(\mathcal{A})$, by the previous corollary, X is coded by $\mathcal{A}$ and hence also by $\mathcal{B}$. □

The next lemma is the analog of Lemma IV.1.5 that says that 1-generics are generalized low. Recall from Exercise II.2.6 that there are ω-presentations $\mathcal{B}$ with $\vec{K}^{\mathcal{B}} <_T D(\mathcal{B})'$.

LEMMA IV.3.4. [**Vat11**] *If $\mathcal{B}$ is 1-generic, then $\vec{K}^{\mathcal{B}} \equiv_T D(\mathcal{B})'$.*

PROOF. We already know that $\vec{K}^{\mathcal{B}} \leq_T D(\mathcal{B})'$ for every presentation $\mathcal{B}$. Let us prove that $\vec{K}^{\mathcal{B}} \geq_T D(\mathcal{B})'$. Take $e \in \mathbb{N}$; we want to decide if $\Phi_e^{D(\mathcal{B})}(e) \downarrow$ using $\vec{K}^{\mathcal{B}}$ as an oracle uniformly in e. Consider the set

$$R_e = \{\bar{q} \in B^\star : \Phi_e^{D_{\mathcal{B}}(\bar{q})}(e) \downarrow\}.$$

Since R_e is r.i.c.e., it is decided by some tuple of the form $\langle 0, ..., k-1\rangle$. The set of tuples which force $g \in [R_e]$, namely

$$\{\bar{p} \in B^\star : \exists \bar{q} \subseteq \bar{p}\ (\bar{q} \in R_e)\},$$

is Σ_1^{c}. The set of tuples which force $g \notin [R_e]$, namely

$$\{\bar{p} \in B^\star : \forall \bar{q} \supseteq \bar{p}\ (\bar{q} \notin R_e)\},$$

is Π_1^{c}. Using $\vec{K}^{\mathcal{B}}$, we can then find such a k and decide whether $\langle 0, ..., k-1\rangle$ forces $g \in [R_e]$ and hence that $e \in D(\mathcal{B})'$, or $\langle 0, ..., k-1\rangle$ forces $g \notin [R_e]$ and hence that $e \notin D(\mathcal{B})'$. □

CHAPTER V

Degree Spectra

Among the main objectives of the field is measuring the computational complexity of structures. There are various ways of doing this. The most common one is through degree spectra.

We already know how to assign a Turing degree to an ω-presentation (namely $D(\mathcal{A})$, as in Subsection I.1.1), but a structure may have many ω-presentations with different Turing degrees. We want a measure of complexity that is independent of the ω-presentation.

DEFINITION V.0.5. The *degree spectrum* of a structure $\mathcal{M}$ is the set

$$DgSp(\mathcal{M}) = \{X \in 2^{\mathbb{N}} : X \text{ computes a copy of } \mathcal{M}\}.$$

Degree spectra are closed upward under Turing reduction. Thus, we can think of them as sets of Turing degrees rather than sets of reals. As it follows from Knight's Theorem I.2.1, $DgSp(\mathcal{A})$ is the set of Turing degrees of the copies of $\mathcal{A}$, provided $\mathcal{A}$ is non-trivial.

Understanding which subsets of the Turing degrees can be realized as degree spectra is an important open question in the area.

V.1. The c.e. embeddability condition

In her Ph.D. thesis [**Ric77**], Linda Richter showed that there are structures which do not have enumeration degree (Definition III.8.2). She gave a general sufficient condition for this to happen:

DEFINITION V.1.1. [**Ric81**, Section 3] A structure $\mathcal{A}$ has the *computable embeddability condition* if each $\exists$-type realized in $\mathcal{A}$ is computable. A structure $\mathcal{A}$ has the *c.e. embeddability condition* if each $\exists$-type realized in $\mathcal{A}$ is c.e.

The reason Richter introduced this notion was to prove Theorem V.1.4 and the corollary below it.

HISTORICAL REMARK V.1.2. *Richter's original definition was not in terms of types, but in terms of finite structures embeddable in $\mathcal{A}$ and extending a fixed tuple, as in the following exercise. Also, she defined the computable embeddability condition and not the c.e. one. However, Theorem V.1.4 below has a more rounded statement when we consider the latter notion. In Russia, structures with the c.e. embeddability condition are called* locally constructivizable.

EXERCISE V.1.3. For each tuple $\bar{a} \in A^{<\mathbb{N}}$, prove that the set

$$\{D_{\mathcal{A}}(\bar{a}\bar{b}) : \bar{b} \in A^{<\mathbb{N}}\} \subseteq 2^{<\mathbb{N}}$$

is "positive-tt equivalent" to $\exists$-$tp_{\mathcal{A}}(\bar{a})$. In particular, they are both Turing and enumeration equivalent. (For the definition of positive-tt reducibility, see page xiii.)

THEOREM V.1.4. *(Richter) Let $\mathcal{A}$ be any structure. The following are equivalent:*
(1) $\mathcal{A}$ has the c.e. embeddability condition.
(2) Every set $X \subseteq \mathbb{N}$ coded by $\mathcal{A}$ is already c.e.

Recall from Section II.1.4 that X is coded by $\mathcal{A}$ if X is c.e. in every presentation of $\mathcal{A}$.

PROOF. To show that (1) implies (2), recall Knight's Theorem II.1.21 that if X is coded by $\mathcal{A}$, it must be enumeration reducible to some $\exists$-type realized in $\mathcal{A}$. Since these are all c.e., X must be c.e. too.

For the other direction, notice that every $\exists$-type realized in $\mathcal{A}$ is coded by $\mathcal{A}$, and hence (2) implies they are all c.e. □

COROLLARY V.1.5. *If $\mathcal{A}$ has the c.e. embeddability condition and has enumeration degree X, then $X \equiv_e \emptyset$.*

COROLLARY V.1.6. *If $\mathcal{A}$ has the c.e. embeddability condition and has Turing degree X, then $X \equiv_T \emptyset$.*

PROOF. Apply the previous corollary to $X \oplus X^c$. □

Richter's original result is actually stronger than Theorem V.1.4. We say that X and Y from a *c.e.-minimal pair* if no set is c.e. in both X and Y, unless it is already c.e.

THEOREM V.1.7. *(Richter) Let $\mathcal{A}$ have the c.e. embeddability condition. Then, for every non-computable set X, there is a copy $\mathcal{B}$ of $\mathcal{A}$ that forms a c.e.-minimal pair with X.*

Notice that a c.e.-minimal pair is also a *minimal pair* in the sense that whenever a set is computable in both X and $D(\mathcal{B})$, it is already computable.

PROOF. Let $\mathcal{B}$ be an X'-1-generic presentation of $\mathcal{A}$. Let Y be c.e. in both X and $D(\mathcal{B})$. Since Y is c.e. in X, $\mathcal{B}$ is Y-1-generic. Then, since Y is c.e. in $D(\mathcal{B})$, Y must be coded by $\mathcal{A}$ (Corollary IV.3.2) and thus be c.e. □

EXAMPLE V.1.8. Richter then showed that linear orderings have the computable embeddability condition. This is because the set of finite extensions of a tuple $a_1, ..., a_k$, namely $\{D_{\mathcal{A}}(\bar{a}\bar{b}) : \bar{b} \in A^{<\mathbb{N}}\} \subseteq 2^{<\mathbb{N}}$, is determined by the ordering among the elements of the tuple, how many elements are in between each pair from the tuple, how many elements are to the left of the whole tuple, and how many are to the right. By "how many," we mean either a finite number or infinity. Thus, a k-type is determined by a permutation σ of $\{1, 2, ..., k\}$, and a $k+1$ tuple from $\mathbb{N} \cup \{\infty\}$. Given that information, one can computably decide if an $\exists$-formula belongs to the type or not.

EXAMPLE V.1.9. Richter also showed that trees in the language of posets also have the computable embeddability condition. We defer this proof to Part 3, where we will prove that the class of trees is Σ-small.

V.2. Co-spectra

The degree spectrum of a structure measures how difficult it is to present the structure. If instead we want to measure how much information is encoded in a structure, the first approach is to use co-spectra. This is not the only approach because, as we will see later, information can be coded within a structure in many different ways — as for instance, it can be coded in the jump of the structure without getting reflected in the co-spectra.

Definition V.2.1. The *co-spectra* of a structure $\mathcal{A}$ is the set

$$\text{co-}DgSp(\mathcal{A}) = \{X \subseteq \mathbb{N} : X \text{ is coded by } \mathcal{A}\}.$$

Recall that X is coded by $\mathcal{A}$ if and only if $\vec{X}$ is r.i.c.e. in $\mathcal{A}$, if and only if $X \leq_e \exists\text{-}tp_{\mathcal{A}}(\bar{p})$ for some $\bar{p} \in A^{<\mathbb{N}}$, and if and only if X is c.e. in every $Y \in DgSp(\mathcal{A})$ (see Section II.1.4). Note that a structure has trivial co-spectrum (i.e., the class of just the c.e. sets) if and only if it has the c.e. embeddability condition.

Definition V.2.2. A set $\mathcal{S} \subseteq \mathcal{P}(\mathbb{N})$ is an *ideal in the enumeration degrees* if it is closed downward under enumeration reducibility and closed under joins.

Co-spectra are always ideals in the enumeration degrees. The reverse is also true.

Lemma V.2.3 (Soskov [**Sos04**]). *Every countable ideal in the enumeration degrees $\mathcal{S} \subseteq \mathcal{P}(\mathbb{N})$ is the co-spectrum of some structure.*

Proof. Given a set X, let $\mathcal{G}_X$ be the graph from Example III.8.3 with one modification: $\mathcal{G}_X$ is made out of cycles of length $n+3$ for $n \in X$, all of these cycles sharing exactly one common node — we call it a *flower graph* because the cycles look like petals coming out of a center node. For a set $\mathcal{S} \subseteq \mathcal{P}(\mathbb{N})$, let $\mathcal{G}_{\mathcal{S}}^{\infty}$ be the graph formed by the disjoint and disconnected union of the graphs $\mathcal{G}_X$ for $X \in \mathcal{S}$, each one repeated infinitely often. Clearly $\mathcal{S} \subseteq \text{co-}DgSp(\mathcal{G}_{\mathcal{S}}^{\infty})$, as for every $X \in \mathcal{S}$, X is c.e. in every copy of $\mathcal{G}_X$. Conversely, we claim that the $\exists$-type of any tuple $\bar{p} \in A^{<\mathbb{N}}$ is e-reducible to a finite join of X's in $\mathcal{S}$. To see this, let $X_1, ..., X_n \in \mathcal{S}$ be such that the elements of $\bar{p}$ are in $\bigcup_{i=1}^{n} \mathcal{G}_{X_i}$. Let $\tilde{\mathcal{G}}$ consist of $\bigcup_{i=1}^{n} \mathcal{G}_{X_i}$ and infinitely many copies of $\mathcal{G}_{\mathbb{N}}$ (i.e., G_Y for $Y = \mathbb{N}$), and let $\bar{q}$ be the tuple in $\tilde{\mathcal{G}}$ corresponding to $\bar{p}$ (i.e., under the isomorphism between the pieces of the form $\bigcup_{i=1}^{n} \mathcal{G}_{X_i}$). We claim that $\exists\text{-}tp_{\mathcal{G}}(\bar{p}) = \exists\text{-}tp_{\tilde{\mathcal{G}}}(\bar{q})$: just observe that there are embeddings $\mathcal{G} \to \tilde{\mathcal{G}}$ (because each $\mathcal{G}_X$ embeds in $\mathcal{G}_{\mathbb{N}}$) and $\tilde{\mathcal{G}} \to \mathcal{G}$ (because $\mathbb{N} \in \mathcal{S}$) matching $\bar{p}$ and $\bar{q}$, and recall that $\exists$-formulas are preserved forward under embeddings. One can easily build a copy of $\tilde{\mathcal{G}}$ from an enumeration of $X_1 \oplus \cdots \oplus X_n$, and hence $\exists\text{-}tp_{\tilde{\mathcal{G}}}(\bar{q}) \leq_e X_1 \oplus \cdots \oplus X_n \in \mathcal{S}$. We conclude that $\exists\text{-}tp_{\mathcal{G}}(\bar{p}) \in \mathcal{S}$. □

Richter's Theorem V.1.7 can be generalized to arbitrary co-spectra as follows.

Lemma V.2.4. *Suppose that every set in co-$DgSp(\mathcal{A})$ is c.e. in Y. Then there is a copy $\mathcal{B}$ of $\mathcal{A}$ such that $D(\mathcal{B})$ and Y are a c.e.-exact pair for co-$DgSp(\mathcal{A})$; that is, such that, for $Z \subseteq \mathbb{N}$, $Z \in$ co-$DgSp(\mathcal{A})$ if and only if Z is c.e. in both $D(\mathcal{B})$ and Y.*

Proof. Let $\mathcal{B}$ be a Y'-1-generic copy of $\mathcal{A}$. Suppose now that X is c.e. in both Y and $D(\mathcal{B})$. Since X is a column in Y', $\mathcal{B}$ is also X-1-generic. Then, by Corollary IV.3.2, X must be coded by $\mathcal{B}$, and hence belongs to co-$DgSp(\mathcal{A})$. □

Recall from Corollary IV.3.3, that we can actually get two copies $\mathcal{B}$ and $\mathcal{C}$ of a structure $\mathcal{A}$ such that a set Z is c.e. in in both if and only if it is in co-$DgSp(\mathcal{A})$. That is, $D(\mathcal{B})$ and $D(\mathcal{C})$ for a c.e.-exact pair for co-$DgSp(\mathcal{A})$.

V.3. Degree spectra that are not possible

In this section, we look at upward closed sets of Turing degrees that cannot be degree spectra.

The first observation along these lines is that degree spectra are always Borel. This will follow from Part 2 where we prove that every structure has a Scott sentence. But among upward closed Borel sets of Turing degrees, we know very little about which ones can be degree spectra and which ones cannot.

V.3.1. No two cones. One of the most best-known results in this vein is due to Knight and her group in the 90's and says that no degree spectrum can be a non-trivial union of two upper cones of Turing degrees — not even the union of countably many upper cones. Her result also applies to the following kind of cone: the *enumeration upper cone with base* X, nameley the set $\{Z \in 2^{\mathbb{N}} : X \text{ is c.e. in } Z\}$.

Theorem V.3.1 (Knight et al.). *No degree spectrum is the union of countably many enumeration upper cones, unless it is equal to just one enumeration upper cone.*

Proof. Suppose that we have $X_1, X_2, ... \subseteq \mathbb{N}$ and a structure $\mathcal{A}$ with

$$DgSp(\mathcal{A}) = \bigcup_{n\in\mathbb{N}} \{Z \in 2^{\mathbb{N}} : X_n \text{ is c.e. in } Z\}.$$

Let $X = \bigoplus_n X_n$. Let $\mathcal{C}$ be a copy of $\mathcal{A}$ such that $\mathcal{C}$ is X-1-generic. Since $D(\mathcal{C}) \in DgSp(A)$, there must be an n such that X_n is c.e. in $D(\mathcal{C})$. From Lemma IV.3.2, we get that X_n is coded by $\mathcal{C}$. But then $DgSp(\mathcal{A}) \subseteq \{Z \in 2^{\mathbb{N}} : X_n \text{ is c.e. in } Z\}$, and hence $DgSp(\mathcal{A}) = \{Z \in 2^{\mathbb{N}} : X_n \text{ is c.e. in } Z\}$ is a single enumeration upper cone. □

Observation V.3.2. No degree spectrum is the union of countably many Turing upper cones, unless it is equal to just one Turing upper cone: To see this, replace X_n by $X_n \oplus X_n^c$ in the proof of the theorem above.

Proof. □

V.3.2. Upward closure of F_σ. We can generalize Observation V.3.2 quite a bit by extending some ideas of U. Andrews and J. Miller [**AM15**]. Recall that we give $\mathbb{N}^{\mathbb{N}}$ and $2^{\mathbb{N}}$ the product topology of the discrete topology on $\mathbb{N}$ and 2 respectively. Thus, the topology on $\mathbb{N}^{\mathbb{N}}$ is generated by the basic open sets $[\sigma] = \{X \in \mathbb{N}^{\mathbb{N}} : \sigma \subset X\}$ for $\sigma \in \mathbb{N}^{<\mathbb{N}}$, and similarly on $2^{\mathbb{N}}$. Open set are then of the form $[R] = \bigcup\{[\sigma] : \sigma \in R\}$ for some $R \subseteq \mathbb{N}^{<\mathbb{N}}$. The complement of $[R]$ can then be viewed as the set of paths through the tree $T = \{\tau \in \mathbb{N}^{<\mathbb{N}} : (\forall \sigma \in R)\sigma \not\subseteq \tau\}$. We thus have that a set $P \subseteq \mathbb{N}^{\mathbb{N}}$ is closed if and only if it is the set of paths $[T]$ through some tree $T \subseteq \mathbb{N}^{<\mathbb{N}}$. One can show that a set is closed if and only if it can be defined as the set of all $X \in \mathbb{N}^{<\mathbb{N}}$ which satisfy some boldface $\mathbf{\Pi}^0_1$-formula $\varphi(X)$ of arithmetic. When $\varphi(X)$ can be taken to be lightface Π^0_1, we say that P is a Π^0_1 *class*. This coincides with the case when T can be taken to be computable.

DEFINITION V.3.3. A set $P \subseteq \mathbb{N}^{<\mathbb{N}}$ is a Π^0_1 *class* if there exists a computable tree T such that $P = [T]$.

Recall that a subset of $\mathbb{N}^{\mathbb{N}}$ is F_σ if it is a countable union of closed sets, or equivalently, if it can be defined by a boldface $\mathbf{\Sigma}^0_2$-formula $\varphi(X)$ of arithmetic. For $F \subseteq \mathbb{N}^{\mathbb{N}}$, we define the *Turing-upward closure of a set* F to be $\{X \in \mathbb{N}^{\mathbb{N}} : \exists Y \in F \ (Y \leq_T X)\}$.

THEOREM V.3.4. *A degree spectrum is never the Turing-upward closure of an F_σ set of reals in $\mathbb{N}^{\mathbb{N}}$, unless it is an enumeration-cone.*

COROLLARY V.3.5 (Knight et al.). *A degree spectrum is never the countable union of countably many Turing-cones, unless it is a single cone.*

PROOF OF COROLLARY. Every countable set is F_σ, so if a degree spectra is the Turing-upper closure of a countable set, it must be an e-cone. But no e-cone is the Turing-upper closure of a countable set unless is the whole $\mathbb{N}^{\mathbb{N}}$: To see this, if $X_0, X_1, ...$ are non-computable and all can compute enumerations of a set Z, then an X-1-generic enumeration of Z, where $X = \bigoplus_n X_n$, is not in the cone above any of the X_n's. □

Another corollary is that the following familiar classes of degrees are not degree spectra: DNC degrees, ML-random degrees, and PA degrees — they are are all F_σ classes of reals. We will get this and a bit more below in Corollary V.3.7, after we prove the following theorem, which contains some of the main ideas to for Theorem V.3.4.

THEOREM V.3.6 (U. Andrews, J. Miller [**AM15**, Proposition 3.9]). *Let $\mathcal{A}$ satisfy the c.e. embeddability condition. Then $\mathcal{A}$ has a copy $\mathcal{B}$ such that, for every Π^0_1 class $P \subseteq \mathbb{N}^{\mathbb{N}}$, $D(\mathcal{B})$ computes no real of P unless P has a computable member already.*

PROOF. Let g be a 1-generic enumeration of $(\mathcal{A}, \vec{K}^{\mathcal{A}})$ and let $\mathcal{B}$ be the 1-generic presentation obtained as the pull-back of $\mathcal{A}$ through g (recall that $\vec{K}^{\mathcal{A}}$ is a complete r.i.c.e. relation on $\mathcal{A}$). Such an enumeration g will be called 2-generic in Part 2. Consider a Π^0_1 class P and let $T \subseteq \mathbb{N}^{<\mathbb{N}}$ be a computable tree with $P = [T]$. Let Φ be a computable operator such that $\Phi^{D(\mathcal{B})}$ is a path through T, i.e., $\Phi^{D(\mathcal{B})}(n) \in \mathbb{N}^n \cap T$ and $\Phi^{D(\mathcal{B})}(n) \subseteq \Phi^{D(\mathcal{B})}(n+1)$ for every n. We need to prove that P has a computable path.

Let us start by forcing $\Phi^{D(\mathcal{B})}$ to output the right kind of values. For this, consider the set of strings which force $\Phi^{D(\mathcal{B})}$ not to:

$$Q_0 = \{\bar{p} \in A^\star : \exists n < |\bar{p}| \ (\Phi^{D_{\mathcal{A}}(\bar{p})}(n) \downarrow \& \ \Phi^{D_{\mathcal{A}}(\bar{p})}(n) \notin \mathbb{N}^n \cap T)\}.$$

The set Q_0 is r.i. computable in $\mathcal{A}$, and hence decided by some initial segment of the enumeration g. No initial segment of g is in Q_0 because $\Phi^{D_{\mathcal{A}}(\mathcal{B})}(n) \in \mathbb{N}^n \cap T$, so there must be an initial segment $\bar{b}_0 \in A^\star$ of the enumeration of $\mathcal{B}$ such that no extension of $\bar{b}_0$ is in Q_0. This means that whenever $\bar{p} \in A^\star$ extends $\bar{b}_0$, if $\Phi^{D_{\mathcal{A}}(\bar{p})}(n) \downarrow$, then $\Phi^{D_{\mathcal{A}}(\bar{p})}(n) \in \mathbb{N}^n \cap T$.

Second, we force the values of $\Phi^{D(\mathcal{B})}$ to be compatible. For this, consider the set of strings which force they are not:

$$Q_1 = \{\bar{p} \in A^\star : \exists n < |\bar{p}| \ (\Phi^{D_{\mathcal{A}}(\bar{p})}(n) \downarrow \& \ \Phi^{D_{\mathcal{A}}(\bar{p})}(n+1) \downarrow \& \ \Phi^{D_{\mathcal{A}}(\bar{p})}(n) \not\subseteq \Phi^{D_{\mathcal{A}}(\bar{p})}(n+1)\}.$$

The set Q_1 is r.i. computable in $\mathcal{A}$, and hence decided by some initial segment of the enumeration g. Again, since $\Phi^{D(\mathcal{B})} \in [T]$, no initial segment of g is in Q_1, and there must be one, $\bar{b}_1 \in A^\star$, none of whose extensions is in Q_1. We may assume $\bar{b}_1 \supseteq \bar{b}_0$.

Third, we force that $\Phi^{D(\mathcal{B})}$ is total: For this, consider the set of strings which force $\Phi^{D(\mathcal{B})}$ to be undefined at some $n \in \mathbb{N}$:

$$Q_2 = \{\bar{p} \in A^\star : \exists n \in \mathbb{N} \forall \bar{q} \in A^\star \left(\bar{q} \supseteq \bar{p} \to \Phi^{D_{\mathcal{A}}(\bar{q})}(n) \uparrow\right)\}.$$

The set Q_2 is Σ^{c}_2 in $\mathcal{A}$, and hence r.i.c.e. in $(\mathcal{A}, \vec{K}^{\mathcal{A}})$ and decided by an initial segment of g. (To see that Q_2 is Σ^{c}_2, observe that $\{\langle \bar{p}, n\rangle : \forall \bar{q} \in A^\star \left(\bar{q} \supseteq \bar{p} \to \Phi^{D_{\mathcal{A}}(\bar{q})}(n) \uparrow\right)\}$ is co-r.i.c.e. in $\mathcal{A}$ and hence Π^{c}_1-definable.) We cannot have an initial segment of g in Q_2 because we would have that $\Phi^{D(\mathcal{B})}(n) \uparrow$ for some n. So, for some initial segment $\bar{b}_2$ of g, we have that for every $\bar{p} \in A^\star$ extending $\bar{b}_2$ and every n, there is a $\bar{q} \in A^\star$ extending $\bar{p}$ for which $\Phi^{D_{\mathcal{B}}(\bar{p})}(n) \downarrow$. We may assume $\bar{b}_2 \supseteq \bar{b}_1$.

Now, using $\exists\text{-}tp_{\mathcal{B}}(\bar{b}_2)$ as a parameter, which we know is c.e., we define a computable path through P. Define a path $\{\sigma_n : n \in \mathbb{N}\} \subseteq T$ step by step as follows. Let σ_0 be the empty string. Given σ_n, chose $\sigma_{n+1} \in \mathbb{N}^{n+1} \cap T$ with $\sigma_{n+1} \supseteq \sigma_n$ to be a string such that

$$\mathcal{A} \models \exists \bar{x}(\Phi^{D_{\mathcal{A}}(\bar{b}_2, \bar{x})}(n+1) = \sigma_{n+1}),$$

or equivalently, such that there exists $\tau \in 2^{<\mathbb{N}}$ with $\Phi^{\tau}(n+1) = \sigma_{n+1}$ for which the formula $\exists \bar{x}(D(\bar{b}_2, \bar{x}) = \tau)$ is in $\exists\text{-}tp_{\mathcal{B}}(\bar{b}_2)$. We know σ_{n+1} exists because, if $\bar{a}_n$ was the witness to define σ_n (i.e., $\Phi^{D_{\mathcal{A}}(\bar{b}_2, \bar{a}_n)}(n) = \sigma_n$), then we know there is an extension $\bar{a}_{n+1}$ of $\bar{a}_n$ such that $\Phi^{D_{\mathcal{A}}(\bar{b}_2, \bar{a}_{n+1})}(n+1) \downarrow$. We also know that $\Phi^{D_{\mathcal{A}}(\bar{b}_2, \bar{a}_{n+1})}(n+1)$ must be in $\mathbb{N}^{n+1} \cap T$ and must extend σ_n. That is our σ_{n+1}. □

Recall that a real $X \in 2^{\mathbb{N}}$ is *diagonal non-computable* (DNC) if $\forall n(X(n) \neq \Phi_n(n))$; a real is ML-random if it does not belong to any effectively-null G_δ set; and a real is PA if it computes a complete theory extending the axioms of Peano arithmetic. See [**Nie09**] for more background on these classes.

Corollary V.3.7 (U. Andrews, J. Miller [**AM15**]). *The class of DNC degrees, the class of ML-random degrees, and the class of PA degrees are not degree spectra. Furthermore, if a structure has the c.e. embeddability property, its degree spectrum is not contained in any of these classes.*

Proof. All these classes are easily seen to be F_σ, and hence they cannot be degree spectra.

Furthermore, The classes of DNC and PA reals are both Π^0_1 classes without computable members, and the class of ML-random reals is an effective countable union of Π^0_1 classes without computable members. So, the second part of the corollary follows form Theorem V.3.6. □

Let us now give the proof of V.3.4, for which we recommend the reader reads the proof of Theorem V.3.6 first.

Proof of Theorem V.3.4. Suppose $\mathcal{A}$ is a structure whose degree spectrum is the Turing-upper closure of an F_σ set $F \subseteq 2^{\mathbb{N}}$. Assume $F = \bigcup_{i \in \mathbb{N}} P_i$ where each $P_i = [T_i]$ for trees $T_i \subseteq 2^{<\mathbb{N}}$. Let g be a $(\bigoplus_{i \in \mathbb{N}} T_i)$-1-generic enumeration of $(\mathcal{A}, \vec{K}^{\mathcal{A}})$ and let $\mathcal{B}$ be the pull-back structure. There is a computable functional Φ and an i such that $\Phi^{D(\mathcal{B})}$

is a path through T_i, i.e., $\Phi^{D(\mathcal{B})}(n) \in \mathbb{N}^n \cap T_i$ and $\Phi^{D(\mathcal{B})}(n) \subseteq \Phi^{D(\mathcal{B})}(n+1)$ for every n. As in the proof of Theorem V.3.6, there is an initial segment $\bar{b} \in A^\star$ of the enumeration g which has no extensions in Q_0, Q_1, and Q_2. That, the tuple $\bar{b}$ satisfies:

(1) $(\forall \bar{q} \supseteq \bar{b}, \bar{q} \in A^\star)$, if $\Phi^{D(\bar{q})}(n)\downarrow$, then $\Phi^{D(\bar{q})}(n) \in T_i \cap 2^{\mathbb{N}}$
(2) $(\forall \bar{q} \supseteq \bar{b}, \bar{q} \in A^\star)$, if $\Phi^{D(\bar{q})}(n)\downarrow \& \ \Phi^{D(\bar{q})}(n+1)\downarrow$, then $\Phi^{D(\bar{q})}(n) \subset \Phi^{D(\bar{q})}(n+1)$.
(3) $(\forall n \in \mathbb{N} \forall \bar{q} \supseteq \bar{b}, \bar{q} \in A^\star)(\exists \bar{p} \supseteq \bar{q}, \bar{p} \in A^\star)\ \Phi^{D(\bar{p})}(n)\downarrow$.

Consider now the tree of possible values of Φ:

$$S = \{\sigma \in 2^{<\mathbb{N}} : (\exists \bar{q} \supseteq \bar{b}, \bar{q} \in A^\star)\ \sigma \subseteq \Phi^{D(\bar{q})}\}.$$

By the assumptions on $\bar{b}$, we get that S is a subtree of T_i without dead ends. From its definition we get hat S is r.i.c.e. in $\mathcal{A}$. On the other hand, every enumeration of S can compute a path through S, and hence a path through T_i, which must then compute a copy of $\mathcal{A}$. Therefore, $\mathcal{A}$ has e-degree S. □

There are enumeration cones which are the Turing upward closure of closed sets, but which are not Turing cones. Furthermore, J. Miller and M. Soskova proved this is the case for all continuous degrees which are not total.

V.4. Some particular degree spectra

We already saw that all upper cones and enumeration cones can be realized as degree spectra (Example III.8.3). In this section, we look at another easy-to-describe but more surprising degree spectra.

V.4.1. The Slaman–Wehner Family. The Slaman–Wehner structure is one that has no computable copy, but is computable in any non-computable set. The best way to describe it is using families of sets.

Definition V.4.1. We say that X can *enumerate a family* $\mathcal{S} \subseteq \mathcal{P}(\mathbb{N})$ if there is an X-c.e. set W such that $\mathcal{S} = \{W^{[n]} : n \in \mathbb{N}\}$.

Observation V.4.2. For every countable family $\mathcal{S} \subseteq \mathcal{P}(\mathbb{N})$, there is a graph $\mathcal{G}_{\mathcal{S}}^{\infty}$ such that, for every oracle X, X can compute a copy of $\mathcal{G}_{\mathcal{S}}^{\infty}$ if and only if X can enumerate $\mathcal{S}$: As in the proof of Lemma V.2.3, consider the *bouquet graph* $\mathcal{G}_{\mathcal{S}}^{\infty} = \bigcup_{Y \in \mathcal{S}, i \in \mathbb{N}} \mathcal{G}_Y$, where $\mathcal{G}_Y$ is the flower graph coding Y, that is $\mathcal{G}_Y$ contains a cycle of length $n+3$ for each $n \in Y$, and all the cycles intersect in one node. Notice that each G_Y appears infinitely often in $\mathcal{G}_S$.

Theorem V.4.3 (Slaman [**Sla98**], Wehner [**Weh98**]). *There is a structure $\mathcal{W}$ whose degree spectrum is $\{X \in 2^{\mathbb{N}} : X \text{ not computable}\}$.*

Proof. Consider the family

$$\mathcal{F} = \{F \oplus \{n\} : F \subseteq \mathbb{N} \text{ finite } \& \ F \neq W_n\},$$

and let $\mathcal{W} = \mathcal{G}_{\mathcal{F}}^{\infty}$ as in the observation above. We claim that X can enumerate $\mathcal{F}$ if and only if X is not computable.

Suppose $\mathcal{F}$ had a computable enumeration. We could then build a function g that, on input n, outputs the c.e. index of a finite set $W_{g(n)}$ with $W_{g(n)} \neq W_n$: just look through the enumeration of $\mathcal{F}$ until you find a column of the form $F \oplus \{n\}$ for some F and output the c.e. index of that F. This contradicts the recursion theorem.

For the other direction, suppose X is not computable. We define an X-computable enumeration of $\mathcal{F}$. Let $Y = X \oplus X^c$, which we know is not c.e. At the beginning of stage t, enumerate into $\mathcal{F}$ all the sets of the form $F \oplus \{n\}$ for all $F \subseteq t$, and all $n < t$. If, among the columns that have been enumerated so far, one is of the form $F \oplus n$ with $F = W_n[t]$ (the stage-t approximation to W_n), we take it as a threat, and we add to F the least element of Y that is not in F already. The idea is that no column can be threatened infinitely often because that would imply that $W_n = F \cup Y$, which we know is not c.e.

More formally: Fix $n \in \mathbb{N}$; we want to enumerate the family $\mathcal{F}_n = \{F : F \subseteq \mathbb{N} \text{ finite } \& \ F \neq W_n\}$ uniformly in n. For each finite set F and every $s \in \mathbb{N}$, we will enumerate a set $R_{F,s}$ with the objective of having

$$\{R_{F,s} : F \subseteq \mathbb{N} \text{ finite}, s \in \mathbb{N}\} = \mathcal{F}_n.$$

We define $R_{F,s}$ by stages as $R_{F,s} = \bigcup_{t\in\mathbb{N}} R_{F,s}[t]$, where each $R_{F,s}[t]$ is finite. For $t \leq s$, let $R_{F,s}[t] = F$. At stage $t+1$, if $R_{F,s}[t] = W_n[t]$, we take it as a threat and let $R_{F,s}[t+1] = R_{F,s}[t] \cup \{y\}$, where y is the least element of $Y \smallsetminus R_{F,s}[t]$. The threats to $R_{F,s}$ must eventually stop, as otherwise we would have $W_n = \bigcup_{t\in\mathbb{N}} R_{F,s}[t] = F \oplus Y$, which is not c.e. Thus, $R_{F,s}$ will end up being finite and not equal to W_n, and hence $R_{F,s}$ belongs to $\mathcal{F}_n$. On the other hand, for every finite set $F \neq W_n$, we have $R_{F,s} = F$ for large enough s: Take s so that $(\forall t > s)\ F \neq W_n[t]$. □

Kaliullin [**Kal08**] showed that the non-Δ^0_2 degrees are a degree spectrum (see Exercise VII.3.7). On the other hand, U. Andrews, M. Cai, I. Kalimullin, S. Lempp, J. Miller, and A. Montalbán showed [**ACK**$^+$] that the class of non-Δ^0_n degrees cannot be a degree spectrum, for $n \geq 3$. It remains open whether the non-arithmetic degrees from a degree spectrum.

CHAPTER VI

Comparing Structures

Another tool for measuring the complexity of an object is to have a way to compare it to other objects. For sets of natural numbers, there are various ways to compare their complexity: Turing reducibility, enumeration reducibility, many-one reducibility, etc. For structures, there are also various ways, the most important ones being Muchnik reducibility, Medvedev reducibility, effective interpretability (also known as Σ-definability), and effective bi-interpretability.

VI.1. Muchnik and Medvedev reducibilities

Let us start by defining these reducibilities on classes of sets:

Definition VI.1.1. A class $\mathcal{R} \subseteq 2^{\mathbb{N}}$ is *Muchnik reducible* to a class $\mathcal{S} \subseteq 2^{\mathbb{N}}$ if every real in $\mathcal{S}$ computes a real in $\mathcal{R}$ [**Muč63**]. If so, we write $\mathcal{R} \leq_w \mathcal{S}$, where the '$w$' stands for "weak," in contrast to the following stronger reducibility. A class $\mathcal{R} \subseteq 2^{\mathbb{N}}$ is *Medvedev reducible* to a class $\mathcal{S} \subseteq 2^{\mathbb{N}}$ if there is a computable operator Φ such that $\Phi^X \in \mathcal{R}$ for every $X \in \mathcal{S}$ [**Med55**]. If so, we write $\mathcal{R} \leq_s \mathcal{S}$.

Here is the idea behind these notions. Suppose we have two problems, $\mathcal{R}$ and $\mathcal{S}$, which involve finding reals with certain properties. Let R and S be the sets of reals which are solutions to $\mathcal{R}$ and $\mathcal{S}$ respectively. For either of the two reductions above, $\mathcal{R}$ reduces to $\mathcal{S}$ if and only if we can produce a solution for R using a solution for S.

Both notions generalize both Turing reducibility and enumeration reducibility: For $X, Y \subseteq \mathbb{N}$, we have that $X \leq_T Y$ if and only if $\{X\} \leq_w \{Y\}$, and also if and only if $\{X\} \leq_s \{Y\}$. We have that $X \leq_e Y$ if and only if the set of enumerations of X (i.e., the set of onto functions $f \colon \mathbb{N} \to X$) is Muchink reducible to the set of enumerations of Y, and also, but less trivially, if and only if the set of enumerations of X is Medvedev reducible to the set of enumerations of Y (Selman [**Sel71**]).

When we are considering countable structures, we apply these reducibilities to the set of their ω-presentations:

Definition VI.1.2. A structure $\mathcal{A}$ is *Muchnik reducible* to a structure $\mathcal{B}$ if every ω-presentation of $\mathcal{B}$ computes an ω-presentation of $\mathcal{A}$ or, more precisely, the atomic diagram of every ω-presentation of $\mathcal{B}$ computes the atomic diagram of an ω-presentation of $\mathcal{A}$. If so, we write $\mathcal{A} \leq_w \mathcal{B}$. A structure $\mathcal{A}$ is *Medvedev reducible* to a structure $\mathcal{B}$ if there is a computable operator Φ such that, for every ω-presentation $\hat{\mathcal{B}}$ of $\mathcal{B}$, $\Phi^{D(\hat{\mathcal{B}})} = D(\hat{\mathcal{A}})$ for some ω-presentation $\hat{\mathcal{A}}$ of $\mathcal{A}$ — we write $\mathcal{A} \leq_s \mathcal{B}$. We denote the respective notions of equivalence by $\equiv_w$ and $\equiv_s$ (i.e., $\mathcal{A} \equiv_w \mathcal{B} \iff \mathcal{A} \leq_w \mathcal{B}$ & $\mathcal{B} \leq_w \mathcal{A}$ and $\mathcal{A} \equiv_s \mathcal{B} \iff \mathcal{A} \leq_s \mathcal{B}$ & $\mathcal{B} \leq_s \mathcal{A}$).

Observation VI.1.3. Muchnik reducibility does not capture any more information than degree spectra:

$$\mathcal{A} \leq_w \mathcal{B} \quad \iff \quad DgSp(\mathcal{A}) \supseteq DgSp(\mathcal{B}).$$

Example VI.1.4. Given linear orderings $\mathcal{A}$ and $\mathcal{B}$ with $\mathcal{A}$ isomorphic to a closed segment $[a, b]_{\mathcal{B}}$ of $\mathcal{B}$, $\mathcal{A} \leq_w \mathcal{B}$.

Example VI.1.5. Given a ring R, $R[x] \leq_s R$.

Example VI.1.6. Given a structure $\mathcal{A}$, there exists a graph $\mathcal{G}_{\mathcal{A}}$ such that $\mathcal{A} \equiv_s \mathcal{G}_{\mathcal{A}}$. We will develop this example later in Section VI.2.2.

Example VI.1.7. For a group $\mathcal{G}$, $\mathcal{G}^{\mathbb{N}} \leq_s \mathcal{G}$, but $\mathcal{G} \not\leq_w \mathcal{G}^{\mathbb{N}}$. Take $\mathcal{G} = \bigoplus_{n\in\mathbb{N}} \mathbb{Z}_{p_n} \oplus \bigoplus_{n\in\mathbb{N}^{\setminus 0'}} \mathbb{Z}_{p_n}$.

These reducibilities form upper-semi-lattices; that is, given structures $\mathcal{A}$ and $\mathcal{B}$, if we define $\mathcal{A} \oplus \mathcal{B}$ by putting together disjoint copies of $\mathcal{A}$ and $\mathcal{B}$ and adding a unary relation A that holds only of the elements in the copy of $\mathcal{A}$, then $\mathcal{A}\oplus\mathcal{B}$ is the least upper bound of $\mathcal{A}$ and $\mathcal{B}$ according to both Muchnik and Medvedev reducibilities. In both cases there is a least degree: If a structure has a computable copy, it reduces to every other structure. Another interesting observation is that there is a least non-computable structure.

Observation VI.1.8. The Slaman–Wehner structure $\mathcal{W}$ from Theorem V.4.3 has no computable copies and is Medvedev reducible to all other structures without computable copies. All we have to observe is that the construction in V.4.3 is uniform in X, i.e., that it produces a computable operator Φ such that, for every non-computable X, Φ^X is the atomic diagram of a copy of $\mathcal{W}$.

The following lemma provides some structural information that we can deduce from having a structure Muchnik or Medvedev reducible to another.

Lemma VI.1.9. *If $\mathcal{A} \leq_w \mathcal{B}$, then for every tuple $\bar{a} \in A^{<\mathbb{N}}$, there is a tuple $\bar{b} \in B^{<\mathbb{N}}$ such that $\exists\text{-}tp_{\mathcal{A}}(\bar{a}) \leq_e \exists\text{-}tp_{\mathcal{B}}(\bar{b})$. If also $\mathcal{A} \leq_s \mathcal{B}$, then $\exists\text{-}Th(\mathcal{A}) \leq_e \exists\text{-}Th(\mathcal{B})$.*

We will show that this lemma is also true for $\Sigma^{\mathsf{c}}_\alpha$ types and theories in Part 2.

Proof. For the first part, suppose that $\mathcal{A} \leq_w \mathcal{B}$ and take $\bar{a} \in A^{<\mathbb{N}}$. Essentially we use that co-$DgSp(\mathcal{A}) \subseteq$ co-$DgSp(\mathcal{B})$: Since $\exists\text{-}tp_{\mathcal{A}}(\bar{a})$ is c.e. in every copy of $\mathcal{A}$, it is also c.e. in every copy of $\mathcal{B}$, and hence it is coded by $\mathcal{B}$. By Knight's Lemma II.1.21, $\exists\text{-}tp_{\mathcal{A}}(\bar{a}) \leq_e \exists\text{-}tp_{\mathcal{B}}(\bar{b})$ for some tuple $\bar{b} \in B^{<\mathbb{N}}$.

Suppose now that $\mathcal{A} \leq_s \mathcal{B}$ via a is a computable operator Φ. For a finite tuple $\sigma \in 2^{<\mathbb{N}}$ approximating (possibly the diagram of $\mathcal{B}$), and for an $\exists$-sentence ψ, we write "$\Phi^\sigma \models \psi$" if the finite diagram output by Φ^σ (possibly approximating $D(\mathcal{A})$) is enough to witness that ψ holds. Notice that since ψ is finitary existential, $\mathcal{A} \models \psi$ if and only if $\Phi^{D(\mathcal{B})\upharpoonright s} \models \psi$ for some $s \in \mathbb{N}$.

To show that $\exists\text{-}Th(\mathcal{A}) \leq_e \exists\text{-}Th(\mathcal{B})$, consider an $\exists$-formula ψ about $\mathcal{A}$. Then $\mathcal{A} \models \psi$ if and only if there is some copy $\tilde{\mathcal{B}}$ of $\mathcal{B}$ such that ψ holds in $\Phi^{D(\tilde{\mathcal{B}})}$. This holds if and only if $\Phi^{D(\tilde{\mathcal{B}})\upharpoonright s} \models \psi$ for some $s \in \mathbb{N}$. Let $\bar{p} \in \mathcal{B}^{<\mathbb{N}}$ be an initial segment of the

presentation $\tilde{\mathcal{B}}$ such that $D_{\mathcal{B}}(\bar{p}) \supseteq D(\tilde{\mathcal{B}}) \upharpoonright s$. We then have that $\mathcal{A} \models \psi$ if and only if $\Phi^{D_{\mathcal{B}}(\bar{p})} \models \psi$ for some $\bar{p} \in B^{<\mathbb{N}}$. Thus,

$$\mathcal{A} \models \psi \iff \Phi^{\sigma} \models \psi \text{ for some } \sigma \subseteq D_{\mathcal{B}}(\bar{p}) \text{ for some } \bar{p} \in B^{<\mathbb{N}}.$$

The set $\{\sigma \in 2^{<\mathbb{N}} : \Phi^{\sigma} \models \psi\}$ is computable, and the formula "$\exists \bar{x}(D(\bar{x}) \supseteq \sigma)$" is answered by $\exists\text{-}Th(\mathcal{B})$. We then get an e-reduction from $\exists\text{-}Th(\mathcal{B})$ to $\exists\text{-}Th(\mathcal{A})$: Given an enumeration of $\exists\text{-}Th(\mathcal{B})$, enumerate ψ into $\exists\text{-}Th(\mathcal{A})$ once you find $\sigma \in 2^{<\mathbb{N}}$ with $\Phi^{\sigma} \models \psi$ a see that "$\exists \bar{x}(D(\bar{x}) \supseteq \sigma)$" is enumerated in $\exists\text{-}Th(\mathcal{B})$. □

So far, Muchnik and Medvedev reducibilities seem to behave in a similar way. However, one of the main differences is that adding constant to the structures does not affect Muchnik reducibility, while the following lemma shows that it does affect Medvedev reducibility.

Lemma VI.1.10. *There are structures $\mathcal{B}$ and $\mathcal{C}$ and $c \in C$ with $\mathcal{B} \leq_s (\mathcal{C}, c)$, but $\mathcal{B} \not\leq_s \mathcal{C}$.*

Notice that $\mathcal{B} \leq_s (\mathcal{C}, c)$ implies $\mathcal{B} \leq_w \mathcal{C}$, and hence this is an example where the Muchnik and Medvedev reducibilities differ.

Proof. Let Z be a non-c.e. set. Consider the following families of sets and their respective bouquet graphs (as in Observation V.4.2):

- $\mathcal{S}_0 = \{F : F \subset \mathbb{N} \text{ finite}\}$ and $\mathcal{A} = \mathcal{G}^{\infty}_{\mathcal{S}_0}$.
- $\mathcal{S}_1 = \{Z\}$ and $\mathcal{B} = \mathcal{G}^{\infty}_{\mathcal{S}_1}$.
- $\mathcal{S}_2 = \mathcal{S}_0 \cup \mathcal{S}_1$ and $\mathcal{C} = \mathcal{G}^{\infty}_{\mathcal{S}_2}$.

The family $\mathcal{S}_0$ has a c.e. enumeration. Thus, $\mathcal{A}$ has a computable copy and $\exists\text{-}Th(\mathcal{A})$ is c.e. The family $\mathcal{S}_1$ does not have a c.e. enumeration. Furthermore, an oracle X can compute an enumeration of $\mathcal{S}_1$ if and only if X can enumerate Z. Thus, $DgSp(\mathcal{B}) = \{X \in 2^{\mathbb{N}} : Z \text{ is c.e. in } X\}$ is the e-cone above Z. The same is true for $\mathcal{C}$: clearly, from a copy of $\mathcal{B}$, we can produce one of $\mathcal{C}$ by attaching a computable copy of $\mathcal{A}$, and given a copy of $\mathcal{C}$, we can produce a copy of $\mathcal{B}$ if we can identify the component of $\mathcal{C}$ that corresponds to $\mathcal{G}_Z$. This implies that if c is the center of the flower corresponding to the component $\mathcal{G}_Z$, we get that $\mathcal{B} \equiv_s (\mathcal{C}, c)$.

However, every finite substructure of $\mathcal{C}$ is isomorphic to some finite substructure of $\mathcal{A}$, and vice versa. Since an $\exists$-formula is true of $\mathcal{A}$ if and only if it is true of some finite substructure of $\mathcal{A}$, this implies that $\exists\text{-}Th(\mathcal{C}) = \exists\text{-}Th(\mathcal{A})$, which is c.e. On the other hand, $\exists\text{-}Th(\mathcal{B})$ can enumerate Z, and hence is not c.e. It follows from Lemma VI.1.9 that $\mathcal{B} \not\leq_s \mathcal{C}$. □

Exercise VI.1.11 (Stuckachev [**Stu07**]). Prove that if a structure $\mathcal{A}$ has Turing degree and $\mathcal{B} \leq_w \mathcal{A}$, then for some tuple $\bar{a} \in A^{<\mathbb{N}}$, $\mathcal{B} \leq_s (\mathcal{A}, \bar{a})$.

Kalimullin [**Kal09**] showed this is not true if we only assume that $\mathcal{A}$ has e-degree.

But the difference between Muchnik and Medvedev reducibility is more than just adding constants, as shown in the corollary below. The theorem before it gives a version of the Slaman–Wehner structure which is computable from every non-computable oracle, but not in a uniform way.

THEOREM VI.1.12 (Faizrahmanov and Kalimullin [**FK**]). *There is a structure $\mathcal{A}$ that has an X-computable copy for every non-computable X, but uniformly. That is, there is no computable single operator Φ such that Φ^X is copy of $\mathcal{A}$ for each non-computable X.*

COROLLARY VI.1.13 (Kalimullin [**Kal09**]). *There are structures $\mathcal{A}$ and $\mathcal{W}$ such that $\mathcal{A} \equiv_w \mathcal{W}$, but $\mathcal{A} \not\leq_s (\mathcal{W}, \bar{w})$ for any tuple $\bar{w} \in W^{<\mathbb{N}}$.*

PROOF OF COROLLARY VI.1.13. The structure $\mathcal{W}$ is the Slaman–Wehner structure from Theorem V.4.3 whose degree spectrum is the non-computable sets and for which there exists a Turing operator that outputs a copy of $\mathcal{W}$ whenever a non-computable set is used as an oracle. Moreover, for any $\bar{w} \in W^{<\mathbb{N}}$, we can produce such an operator that outputs a copy of $(\mathcal{W}, \bar{w})$: Recall that $\mathcal{W} = \bigcup_{n\in\mathbb{N}} \mathcal{W}_n$, where $\mathcal{W}_n$ is the disjoint union of the flower graphs $\mathcal{G}_{F\oplus\{n\}}$ for $F \subset \mathbb{N}$ finite with $F \neq W_n$, each appearing infinitely often. There are finitely many components $\mathcal{W}_n$ which contain an element of $\bar{w}$, so we can fix a computable enumeration of them. For the other n's, we can use the construction of Theorem V.4.3.

The structure $\mathcal{A}$ is the one from Theorem VI.1.12. It is Muchnik equivalent to $\mathcal{W}$: it is computable from any non-computable oracle, and it is has no computable copies, as otherwise there would be would be a computable operator that produces a computable ω-presentation of $\mathcal{A}$ ignoring the oracle. $\mathcal{A}$ is not Medvedev reducible to $(\mathcal{W}, \bar{w})$ for any $\bar{w} \in W^{<\mathbb{N}}$ because if there was a computable operator that produces a copy of $\mathcal{A}$ out of every copy of $(\mathcal{W}, \bar{w})$, we could produce a copy of $(\mathcal{W}, \bar{w})$ and then one of $\mathcal{A}$ uniformly from a non-computable set. □

PROOF OF THEOREM VI.1.12. We modify Wehner's construction from Theorem V.4.3. We still consider a family of finite sets of the form $F \oplus \{n\}$, but the difference with Wehner's construction is that we think of F as a finite subset of $\mathbb{Q}$ instead of $\mathbb{N}$, and instead of requiring F to be different from the n-th c.e. set, we just require its maximum to be different from the maximum of the n-th c.e. subset of $\mathbb{Q}$. It works.

Let $\{Q_n : n \in \mathbb{N}\}$ be an effective enumeration of the c.e. subsets of $\mathbb{Q}$. (For example, given an effective Gödel numbering $q \mapsto \ulcorner q \urcorner \colon \mathbb{Q} \to \mathbb{N}$, let $Q_n = \{q \in \mathbb{Q} : \ulcorner q \urcorner \in W_n\}$.) Consider the family of sets

$$\mathcal{F} = \{F \oplus \{n\} : F \subseteq \mathbb{Q} \text{ finite}, n \in \mathbb{N}, \max(F) \neq \max(Q_n)\},$$

where the formula $\max(F) \neq \max(Q_n)$ is assumed to be vacuously true when Q_n does not have a greatest element. Let $\mathcal{A}$ be the associated bouquet graph $\mathcal{G}^\infty_{\mathcal{F}}$ as in Observation V.4.2. Recall that the existence of an X-computable presentation of $\mathcal{G}^\infty_{\mathcal{F}}$ is equivalent to the existence of an X-c.e. enumeration of $\mathcal{F}$, that is, an X-c.e. set V with $\mathcal{F} = \{V^{[n]} : n \in \mathbb{N}\}$.

First, let us show that $\mathcal{F}$ is computably enumerable in every non-computable set X. A real is said to be *left c.e.* if it is of the form $\sup(Q_e)$ for some c.e. set $Q_e \subseteq \mathbb{Q}$. Let α be X-left c.e., but not left c.e. To see that such an α exists, consider $\beta_0 = \sum_{i\in X} 2^{-i}$ and $\beta_1 = \sum_{i\notin X} 2^{-i}$. They cannot be both left c.e., as otherwise X would be computable. Let α be whichever of β_0 or β_1 is not left c.e. — this is the only step in the construction that is not uniform in X. Let $\{\alpha_i : i \in \mathbb{N}\} \subseteq \mathbb{Q}$ be an X-computable increasing sequence with limit α.

Fix n. We want to enumerate the family

$$\mathcal{F}_n = \{F : F \subseteq \mathbb{Q} \text{ finite}, \max(F) \neq \max(Q_n)\}$$

uniformly in n. The idea is to enumerate a new component of the form F for each finite set $F \subseteq \mathbb{Q}$ at each stage, and if, at a certain stage t, we are threatened by having $\max(F) = \max(Q_{n,t})$, we add $\max(F) + \alpha_t$ to that component changing its maximum value. A component can not be threatened infinitely often because we would end up having $\sup(Q_n) = \max(F) + \alpha$, which is not left-c.e. Let us explain this in more detail. For each finite set $F \in \mathbb{Q}$ and $s \in \mathbb{N}$, we will enumerate a set $R_{F,s}$ uniformly in X, with the objective of getting

$$\mathcal{F}_n = \{R_{F,s} : F \subseteq_f \mathbb{Q}, s \in \mathbb{N}\}.$$

The idea is that $R_{F,s}$ starts by being F at stage s and then every time it is threaten, we add a new element to $R_{F,s}$ so as to change its maximum value. To define $R_{F,s}$, we will define a non-decreasing sequence $\{r_{F,s}[t] : t \in \mathbb{N}\} \subset \mathbb{Q}$ and then let

$$R_{F,s} = F \cup \{r_{F,s}[t] : t \in \omega\}.$$

Let $r_{F,s}[t] = \max(F)$ for all $t \leq s$. At stage $t + 1 > s$, if $r_{F,s}[t] = \max(Q_{n,t})$, let $r_{F,s}[t + 1] = \max(F) + \alpha_t$, where $Q_{n,t}$ and α_t are the stage-t approximation to Q_n and α. We claim that this sequence eventually stabilizes. Otherwise, we would have that

$$\sup(Q_n) = \lim_t \max(Q_{n,t}) = \lim_t r_{F,s}[t] = \max(F) + \alpha,$$

contradicting that $\max(F) + \alpha$ is not left c.e. Let $r_{F,s} = \lim_t r_{F,s}[t]$. Then $r_{F,s} \neq \max(Q_n)$ and $R_{F,s} \in \mathcal{F}_n$. On the other hand, for every finite $F \subseteq \mathbb{Q}$ for which $\max(F) \neq \max(Q_n)$, we have that $R_{F,s} = F$ for large enough s: Take s so that $\max(Q_{n,t}) \neq \max(F)$ for all $t > s$, and hence so that $r_{F,s}[t] = \max(F)$ for all $t \in \mathbb{N}$.

For the second part of the theorem, let us assume that V is a c.e. operator such that V^X is an enumeration of $\mathcal{F}$ for every non-computable X, and let us try to get a contradiction. For this, we will define a uniformly c.e. sequence M_n of finite subsets of $\mathbb{Q}$ with $\max(M_n) \neq \max(\mathcal{Q}_n)$. This will give us a contradiction because, if f is a computable function such that $Q_{f(n)} = M_n$, then by the recursion theorem, there must be an n_0 with $W_{f(n_0)} = W_{n_0}$, and hence with $M_{n_0} = Q_{f(n_0)} = Q_{n_0}$.

Using the operator V, we can easily produce a uniform family of c.e. operators $\{U_n : n \in \mathbb{N}\}$ such that $U_n^X \subseteq \mathbb{Q}$ is finite and $\max(U_n^X) \neq \max(Q_n)$ for all non-computable X and $n \in \mathbb{N}$: Search for a column of V^X of the form $F \oplus \{n\}$ for some F (i.e., a column that contains the number $2n + 1$), and let $U_n^X = F$.

For $X \in 2^{\mathbb{N}}$, let

$$m_n^X = \sup(U_n^X) \in \mathbb{R} \cup \{\infty\},$$

which we know is actually a maximum in $\mathbb{Q}$ when X is non-computable. For $\sigma \in 2^{<\mathbb{N}}$, let $m_n^\sigma = \max(U_n^\sigma) \in \mathbb{Q} \cup \{-\infty\}$, where U_n^σ is the step-$|\sigma|$ approximation to U_n^X for $X \supset \sigma$, and where $m_n^\sigma = -\infty$ if $U_n^\sigma = \emptyset$. We have the following properties:

- $\sigma \subseteq \tau \Rightarrow m_n^\sigma \leq m_n^\tau$.
- If $X \in 2^{\mathbb{N}}$ is non-computable, then $m_n^X = m_n^{\sigma_X}$ for some finite $\sigma_X \subset X$.
- If $X \in 2^{\mathbb{N}}$ is non-computable, $m_n^X \neq \max(Q_n)$.

Let $T \subseteq 2^{<\mathbb{N}}$ be a computable tree with no computable paths. (For instance, let $T = \{\sigma \in 2^{<\mathbb{N}} : \forall e < |\sigma| \ (\sigma(e) \neq \Phi_{e,|\sigma|}(e))\}$ whose paths are the 2-DNC reals.) The

idea is to use T to define M_n so that its maximum element is the minimum value of m_n^X among all the $X \in [T]$. Since such $X \in [T]$ would be non-computable, we would have that $\max(M_n) = m_n^X \neq \max(Q_n)$. Let

$$\gamma = \inf\{m_n^X : X \in [T]\} \in \mathbb{R} \cup \{-\infty\};$$

we will show that γ is actually a minimum. Consider the following sequence approximating γ:

$$\gamma[k] = \min(m_n^\sigma : \sigma \in T \cap 2^k).$$

Since $m_n^X \geq m_n^{X\restriction k}$ for all X and k, we get that $\gamma \geq \gamma_k$. To see that the sequence converges to γ, for $\epsilon > 0$ one can find X so that $\gamma - \epsilon < m_n^X = m_n^{\sigma_X} \leq \gamma[|\sigma_X|]$. First, we claim that this sequence becomes constant from some point on. To see this, let us observe that the sub-tree $\{\sigma \in T : m_n^\sigma < \gamma\}$ must be finite: Otherwise, by König's lemma, it would have a path $Y \in [T]$. But then $m_n^Y = m_n^{\sigma_Y} < \gamma$ contradicting the definition of γ. So if k_0 bounds the lengths of all the strings in that tree, $\gamma[k] = \gamma$ for all $k \geq k_0$.

Second, we claim that $\gamma = m_n^X$ for some $X \in [T]$. To see this, let us observe that the tree $\{\sigma \in T : m_n^\sigma \leq \gamma\}$ must have a path: Otherwise, by König's lemma, the tree would be finite, and if k_0 bounds the lengths of all the strings in that tree, we would get $\gamma[k] > \gamma$, which we know does not happen. So, if X is a path through that tree, $m_n^X = m_n^{\sigma_X} \leq \gamma$ and hence m_n^X is minimum among all $X \in [T]$. It follows that $\gamma = m_n^X \neq \max(Q_n)$.

Finally, let

$$M_n = \{\gamma[k] : k \in \mathbb{N}\}.$$

Then M_n must be finite and have maximum element $\gamma \neq \max(Q_n)$. It is not hard to see that $\{M_n : n \in \omega\}$ is c.e. uniformly in n. This finishes the construction of M_n and the proof that $\mathcal{A}$ cannot be uniformly computed from all non-computable sets. □

VI.2. Computable functors and effective interpretability

There is a third important notion of reducibility which has many more structural consequences — it even has a structural characterization in terms of interpretations. It comes from requiring that a Medvedev reduction Φ preserve isomorphisms effectively.

Definition VI.2.1 (R. Miller, B. Poonen, H. Schoutens, and A. Shlapentokh [**MPSS**, Definition 3.1]). Given structures $\mathcal{A}$ and $\mathcal{B}$, a *computable functor* from $\mathcal{B}$ to $\mathcal{A}$ consists of two computable operators, Φ and Ψ, such that:

(1) Φ is a Medvedev reduction witnessing $\mathcal{A} \leq_s \mathcal{B}$; that is, for every copy $\hat{\mathcal{B}}$ of $\mathcal{B}$, $\Phi^{D(\hat{\mathcal{B}})}$ is the atomic diagram of a copy of $\mathcal{A}$.
(2) For every isomorphism f between two copies $\hat{\mathcal{B}}$ and $\tilde{\mathcal{B}}$ of $\mathcal{B}$, $\Psi^{D(\hat{\mathcal{B}}),f,D(\tilde{\mathcal{B}})}$ is an isomorphism between the copies of $\mathcal{A}$ obtained from $\Phi^{D(\hat{\mathcal{B}})}$ and $\Phi^{D(\tilde{\mathcal{B}})}$.

We also require that the operator Ψ preserve the identity and composition of isomorphisms:

(3) $\Psi^{D(\hat{\mathcal{B}}),id,D(\hat{\mathcal{B}})} = id$ for every copy $\hat{\mathcal{B}}$ of $\mathcal{B}$.
(4) $\Psi^{D(\hat{\mathcal{B}}),g\circ f,D(\check{\mathcal{B}})} = \Psi^{D(\check{\mathcal{B}}),g,D(\tilde{\mathcal{B}})} \circ \Psi^{D(\hat{\mathcal{B}}),f,D(\check{\mathcal{B}})}$, for copies $\hat{\mathcal{B}}$, $\tilde{\mathcal{B}}$ and $\check{\mathcal{B}}$ of $\mathcal{B}$ and isomorphisms $f\colon \hat{\mathcal{B}} \to \check{\mathcal{B}}$ and $g\colon \check{\mathcal{B}} \to \tilde{\mathcal{B}}$.

The pair Φ, Ψ is actually a functor in the sense of category theory. It is a functor from the category of ω-presentations of $\mathcal{B}$ where morphisms are the isomorphisms between the copies of $\mathcal{B}$, to the category of ω-presentations of $\mathcal{A}$.

EXAMPLE VI.2.2. Let $\mathcal{B}$ be an integral domain (i.e., a commutative ring without zero-divisors) and let $\mathcal{A}$ be the field of fractions of $\mathcal{B}$. That is, $\mathcal{A}$ consists of element of the form $\frac{p}{q}$ for $p, q \in B$, $q \neq 0$. Equivalence, addition, and multiplication of fractions is defined as usual. One can easily build a computable functor that produces a copy of $\mathcal{A}$ out of a copy of $\mathcal{B}$ and maps isomorphisms between copies of $\mathcal{B}$ to the respective copies of $\mathcal{A}$. We let the reader check the details. We will develop this example further below in Example VI.2.4.

We will prove that having a computable functor is equivalent to having an effective interpretation. Informally, a structure $\mathcal{A}$ is *effectively-interpretable* in a structure $\mathcal{B}$ if there is an interpretation of $\mathcal{A}$ in $\mathcal{B}$ as in model theory, but where the domain of the interpretation is allowed to be a subset of $B^{<\mathbb{N}}$ instead of just B^n, and where all sets in the interpretation are required to be "effectively definable" instead of elementary first-order definable.

Before giving the formal definition, we need to review one more concept. Recall that a relation R on $\mathcal{A}^{<\mathbb{N}}$ is *uniformly r.i.c.e. (u.r.i.c.e.)* if there is a c.e. operator W such that $R^{\mathcal{B}} = W^{D(\mathcal{B})}$ for every copy $(\mathcal{B}, R^{\mathcal{B}})$ of $(\mathcal{A}, R)$. These are exactly the Σ^{c}_1-definable relations without parameters (Corollary II.1.16). Analogously, R is *uniformly r.i. computable* if there is a computable operator Φ such that $R^{\mathcal{B}} = \Phi^{D(\mathcal{B})}$ for every copy $(\mathcal{B}, R^{\mathcal{B}})$ of $(\mathcal{A}, R)$.

DEFINITION VI.2.3. Let $\mathcal{A}$ be a τ-structure, and $\mathcal{B}$ be any structure. Let us assume that τ is a relational vocabulary $\tau = \{P_i : i \in I\}$ where P_i has arity $a(i)$. So $\mathcal{A} = (A; P_0^{\mathcal{A}}, P_1^{\mathcal{A}}, ...)$ and $P_i^{\mathcal{A}} \subseteq A^{a(i)}$.

We say that $\mathcal{A}$ is *effectively-interpretable* in $\mathcal{B}$ if, in $\mathcal{B}$, there are u.r.i. computable relations $A^{\mathcal{B}}$, $\sim^{\mathcal{B}}$, and $\{R_i^{\mathcal{B}} : i \in I\}$ such that

- $A^{\mathcal{B}} \subseteq \mathcal{B}^{<\mathbb{N}}$ (the domain of the interpretation of $\mathcal{A}$ in $\mathcal{B}$),
- $\sim^{\mathcal{B}} \subseteq A^{\mathcal{B}} \times A^{\mathcal{B}}$ is an equivalence relation on $A^{\mathcal{B}}$ (interpreting equality),
- each $R_i^{\mathcal{B}} \subseteq (A^{\mathcal{B}})^{a(i)}$ is closed under the equivalence $\sim^{\mathcal{B}}$ (interpreting the relations P_i),

and there is a function $f_{\mathcal{A}}^{\mathcal{B}} \colon A^{\mathcal{B}} \to A$ which induces an isomorphism:

$$(A^{\mathcal{B}}/\sim^{\mathcal{B}}; R_0^{\mathcal{B}}, R_1^{\mathcal{B}}, ...) \cong (A; P_0^{\mathcal{A}}, P_1^{\mathcal{A}}, ...).$$

Let us clarify this last line. The function $f_{\mathcal{A}}^{\mathcal{B}} \colon A^{\mathcal{B}} \to A$ must be an onto map such that $f_{\mathcal{A}}^{\mathcal{B}}(\bar{a}) = f_{\mathcal{A}}^{\mathcal{B}}(\bar{b}) \iff (\bar{a}, \bar{b}) \in \sim^{\mathcal{B}}$ and $f_{\mathcal{A}}^{\mathcal{B}}(\bar{a}) \in P_i^{\mathcal{A}} \iff \bar{a} \in R_i^{\mathcal{B}}$ for all $\bar{a}, \bar{b} \in (A^{\mathcal{B}})^{<\mathbb{N}}$. Notice that there is no restriction on the complexity or definability of $f_{\mathcal{A}}^{\mathcal{B}}$. We use $\mathcal{A}^{\mathcal{B}}$ to denote the structure $(A^{\mathcal{B}}/\sim^{\mathcal{B}}; R_0^{\mathcal{B}}, R_1^{\mathcal{B}}, ...)$.

If we add parameters, this notion is equivalent to that of Σ-definability, which was introduced by Ershov [**Ers96**] and is widely studied in Russia. Ershov's definition is quite different in format: it uses $\mathrm{HF}(\mathcal{B})$ instead of $\mathcal{B}^{<\mathbb{N}}$ (see Section II.4.1) and sets that are $\exists$-definable over $\mathrm{HF}(\mathcal{B})$ instead of Σ^{c}_1-definable subsets of $B^{<\mathbb{N}}$ (which we know are equivalent; Theorem II.4.3).

EXAMPLE VI.2.4. Recall Example VI.2.2 above. We claim that $\mathcal{A}$ is effectively interpretable in $\mathcal{B}$. Let $A^B = \{(p,q) \in B^2 : q \neq 0\}$. Let $(p_0,q_0) \sim^B (p_1,q_1)$ if $p_0 \times^{\mathcal{B}} q_1 = p_1 \times^B q_0$. Let $((p_0,q_0),(p_1,q_1),(p_2,q_2))$ be in the graph of addition for $\mathcal{A}^{\mathcal{B}}$ if $(p_0 \times^B q_1 +^B p_1 \times^B q_0) \times^B q_2 = q_0 \times^B q_1 \times^B p_2$. Let $((p_0,q_0),(p_1,q_1),(p_2,q_2))$ be in the graph of multiplication for $\mathcal{A}^{\mathcal{B}}$ if $p_0 \times^B p_1 \times^B q_2 = q_0 \times^B q_1 \times^B p_2$.

LEMMA VI.2.5. *An effective interpretation of $\mathcal{A}$ in $\mathcal{B}$ induces a computable functor from $\mathcal{B}$ to $\mathcal{A}$.*

PROOF. Since $A^{\mathcal{B}}$, $\sim^{\mathcal{B}}$, and $\{R_i^{\mathcal{B}} : i \in I\}$ are u.r.i. computable in $\mathcal{B}$, we have a computable operator that gives us those sets within any copy of $\hat{\mathcal{B}}$, using $D(\hat{\mathcal{B}})$ as an oracle. We thus have a computable operator Φ that, given $\hat{\mathcal{B}} \cong \mathcal{B}$, outputs $D(\mathcal{A}^{\hat{\mathcal{B}}})$, the atomic diagram of the congruence ($\subseteq \mathbb{N}^{<\mathbb{N}}$)-presentation $\mathcal{A}^{\hat{\mathcal{B}}}$ of $\mathcal{A}$ with domain $A^{\hat{\mathcal{B}}} \subseteq \hat{B}^{<\mathbb{N}} = \mathbb{N}^{<\mathbb{N}}$. Fixing a bijection between $\mathbb{N}$ and $\mathbb{N}^{<\mathbb{N}}$, and using Lemma I.1.9, we get a computable operator Υ transforming congruence ($\subseteq \omega^{<\mathbb{N}}$)-presentations into injective ω-presentations. Both of these computable operators Φ and Υ preserve isomorphisms effectively; in other words, they can be easily made into computable functors. Composing these computable functors, $\Upsilon \circ \Phi$, we get the computable functor we wanted. □

The following theorem shows that the reversal is also true. Furthermore, given a computable functor, we can get an effective interpretation that essentially induces the original functor back, up to effective isomorphism of functors.

THEOREM VI.2.6 (Harrison-Trainor, Melnikov, Miller, Montalbán [**HTMMM**]). *Let $\mathcal{A}$ and $\mathcal{B}$ be countable structures. The following are equivalent:*

(1) *$\mathcal{A}$ is effectively interpretable in $\mathcal{B}$.*
(2) *There is a computable functor from $\mathcal{B}$ to $\mathcal{A}$.*

We will prove this theorem in Part 2 once we have developed more forcing techniques. The original proof from [**HTMMM**] does not use forcing, and the reader should be able to follow it with what we have learned so far. The proof using forcing [**HTMM**] is much more informative and can be generalized to a broader setting.

VI.2.1. Effective bi-interpretability. Effective interpretability and Σ-definability induce notions of equivalence between structures as usual: two structures are equivalent if they are reducible to each other. Σ-equivalence, the equivalence notion that comes from Σ-definability, has been widely studied. However, it still does not really capture the idea of two structures being "the same from a computability viewpoint." In this section, we introduce the more recent notion of effectively-bi-interpretability, which is a strengthening of Σ-equivalence. For this strengthening, we require the composition of the isomorphisms, interpreting one structure inside the other and then interpreting the other back into the first one, to be effective. We will show how most computability theoretic properties are preserved under this equivalence, and how it applies to examples of structures that we intuitively thought of as being "the same." Here is the formal definition:

DEFINITION VI.2.7. [**Mond**, Definition 5.1] Two structures, $\mathcal{A}$ and $\mathcal{B}$, are *effectively-bi-interpretable* if there are effective-interpretations of each structure in the other as in

Definition VI.2.3 such that the compositions

$$f^{\mathcal{A}}_{\mathcal{B}} \circ \tilde{f}^{\mathcal{B}}_{\mathcal{A}} \colon B^{\mathcal{A}^{\mathcal{B}}} \to B \quad \text{and} \quad f^{\mathcal{B}}_{\mathcal{A}} \circ \tilde{f}^{\mathcal{A}}_{\mathcal{B}} \colon A^{\mathcal{B}^{\mathcal{A}}} \to A$$

are u.r.i. computable in $\mathcal{B}$ and $\mathcal{A}$, respectively.

Let us explain this messy notation. $B^{\mathcal{A}^{\mathcal{B}}} \subseteq (A^{\mathcal{B}})^{<\mathbb{N}} \subseteq (B^{<\mathbb{N}})^{<\mathbb{N}}$ is the domain of the interpretation of $\mathcal{B}$ within the interpretation of $\mathcal{A}$ within $\mathcal{B}$, and $\tilde{f}^{\mathcal{B}}_{\mathcal{A}} \colon (A^{\mathcal{B}})^{<\mathbb{N}} \to A^{<\mathbb{N}}$ is the obvious extension of $f^{\mathcal{B}}_{\mathcal{A}} \colon A^{\mathcal{B}} \to A$ from elements to tuples: $\tilde{f}^{\mathcal{B}}_{\mathcal{A}}(a_0, ..., a_k) = (f^{\mathcal{B}}_{\mathcal{A}}(a_0), ..., f^{\mathcal{B}}_{A}(a_k))$. Notice that since $f^{\mathcal{A}}_{\mathcal{B}} \circ \tilde{f}^{\mathcal{B}}_{\mathcal{A}}$ is a partial function from $(B^{<\mathbb{N}})^{<\mathbb{N}}$ to B, it can be coded by a relation on $B^{<\mathbb{N}}$, and it makes sense to require it to be u.r.i. computable.

Let us make a quick comment on non-relational vocabularies. We have not defined bi-interpretability for non-relational languages, but when the interpretations are injective, Definition VI.2.7 goes through without problems.

In the next lemma, we see how effective-bi-interpretability preserves most computability theoretic properties. Some of the properties we list will not be introduced until later in the book, and we delay their proofs until then.

LEMMA VI.2.8. *Let $\mathcal{A}$ and $\mathcal{B}$ be effectively-bi-interpretable.*
(1) $\mathcal{A}$ and $\mathcal{B}$ have the same degree spectrum.
(2) $\mathcal{A}$ is $\exists$-atomic if and only if $\mathcal{B}$ is.
(3) $\mathcal{A}$ is rigid if and only if $\mathcal{B}$ is.
(4) The automorphism groups of $\mathcal{A}$ and $\mathcal{B}$ are isomorphic.
(5) $\mathcal{A}$ is computably categorical if and only if $\mathcal{B}$ is.
(6) $\mathcal{A}$ and $\mathcal{B}$ have the same computable dimension.
(7) $\mathcal{A}$ has the c.e. extendibility condition if and only if $\mathcal{B}$ does.
(8) The index sets of $\mathcal{A}$ and $\mathcal{B}$ are Turing equivalent, provided $\mathcal{A}$ and $\mathcal{B}$ are infinite.

(Of course, items (5), (6), and (8) assume $\mathcal{A}$ and $\mathcal{B}$ are computable.)

PROOF. Throughout this proof, assume that $\mathcal{A}$ is already the presentation $\mathcal{A}^{\mathcal{B}}$ that is coded inside $\mathcal{B}^{<\mathbb{N}}$, i.e., with domain $A^{\mathcal{B}}$, and $\tilde{\mathcal{B}}$ is the copy of $\mathcal{B}$ coded inside $\mathcal{A}^{<\mathbb{N}}$, i.e., with domain $\mathcal{B}^{\mathcal{A}} = \mathcal{B}^{\mathcal{A}^{\mathcal{B}}}$. We let f be the isomorphism from $\tilde{\mathcal{B}}$ to $\mathcal{B}$.

For part (1), recall from Lemma VI.2.5 that there are computable functors between $\mathcal{A}$ and $\mathcal{B}$, and in particular, that they are Medvedev equivalent, and hence also Muchnik equivalent.

For part (2), suppose $\mathcal{A}$ is $\exists$-atomic, and hence that every automorphism orbit in $\mathcal{A}$ is $\exists$-definable. Take a tuple $\bar{b} \in \mathcal{B}^{<\mathbb{N}}$; we will show its orbit is also $\exists$-definable. Let $\bar{c} \in (B^{\mathcal{A}^{\mathcal{B}}})^{<\mathbb{N}} \subseteq (B^{<\mathbb{N}})^{<\mathbb{N}}$ be such that $f(\bar{c}) = \bar{b}$. The orbit of $\bar{c}$ is $\exists$-definable inside $\mathcal{A}^{\mathcal{B}}$, and since $\mathcal{A}^{\mathcal{B}}$ is $\Delta^{\mathfrak{c}}_1$-definable in $\mathcal{B}$, the orbit of $\bar{c}$ is also $\Sigma^{\mathfrak{c}}_1$ definable in $\mathcal{B}$. Since f is $\Sigma^{\mathfrak{c}}_1$-definable in $\mathcal{B}$, the orbit of $\bar{b}$ is also $\Sigma^{\mathfrak{c}}_1$ definable. If an orbit is definable by a disjunction, it must be defined by one of its disjuncts, and hence the orbit of $\bar{b}$ is $\exists$-definable in $\mathcal{B}$. It follows that $\mathcal{B}$ is $\exists$-atomic.

For part (3), suppose $\mathcal{B}$ is not rigid, and let h be a nontrivial automorphism of $\mathcal{B}$. It then induces an automorphism of $\mathcal{B}^{<\mathbb{N}}$, which then induces an automorphism g_h of $\mathcal{A}^{\mathcal{B}}$, which then induces an automorphism h_{g_h} of $\mathcal{B}^{\mathcal{A}^{\mathcal{B}}}$. Since $f \colon \mathcal{B}^{\mathcal{A}^{\mathcal{B}}} \to \mathcal{B}$ is u.r.i. computable, it is invariant; that is, $f(\bar{a}) = b \iff f(h_{g_h}(\bar{a})) = h(b)$. In other words,

$f \circ h_{g_h} = h \circ f$, and since h is nontrivial, h_{g_h} must be nontrivial too. It follows that the automorphism g_h of $\mathcal{A}$ cannot be trivial either.

For part (4), notice that the composition of the following three maps is the identity on $Aut(\mathcal{B})$: first the homomorphism $h \mapsto g_h \colon Aut(\mathcal{B}) \to Aut(\mathcal{A}^{\mathcal{B}})$; second the homomorphism $g \mapsto h_g \colon Aut(\mathcal{A}^{\mathcal{B}}) \to Aut(\mathcal{B}^{\mathcal{A}^{\mathcal{B}}})$; and third the conjugation isomorphism induced by f, namely $\hat{h} \mapsto f \circ \hat{h} \circ f^{-1} \colon Aut(\mathcal{B}^{\mathcal{A}^{\mathcal{B}}}) \to \mathcal{B}$. We thus get that they they are all isomorphisms.

For part (5), we need the following observation. Let $\mathcal{B}_1$ and $\mathcal{B}_2$ be copies of $\mathcal{B}$. The point we need to make here is that if $\mathcal{A}^{\mathcal{B}_1}$ and $\mathcal{A}^{\mathcal{B}_2}$ are computably isomorphic, then so are $\mathcal{B}_1$ and $\mathcal{B}_2$: A computable isomorphism between $\mathcal{A}^{\mathcal{B}_1}$ and $\mathcal{A}^{\mathcal{B}_2}$ induces a computable isomorphism between $\mathcal{B}^{\mathcal{A}^{\mathcal{B}_1}}$ and $\mathcal{B}^{\mathcal{A}^{\mathcal{B}_2}}$, each of which is computably isomorphic to $\mathcal{B}_1$ or $\mathcal{B}_2$, respectively. Thus, if $\mathcal{A}$ is computably categorical, so is $\mathcal{B}$. For (6), we have that if $\mathcal{B}$ has k non-computably isomorphic copies $\mathcal{B}_1, ..., \mathcal{B}_k$, then the respective structures $\mathcal{A}^{\mathcal{B}_1}, ..., \mathcal{A}^{\mathcal{B}_k}$ cannot be computably isomorphic either. So the effective dimension of $\mathcal{A}$ is at least that of $\mathcal{B}$, and hence, by symmetry, they must be equal.

For part (7), refer to Observation VI.1.3.

In part (8), by the *index set* of a structure $\mathcal{A}$, we mean the set of all i such that W_i is the atomic diagram of a structure isomorphic to $\mathcal{A}$. To prove (8), suppose we are given an index of a computable structure $\mathcal{C}$, and we want to decide if it is isomorphic to $\mathcal{B}$ using the index set of $\mathcal{A}$. Using the formulas in the effective interpretation of $\mathcal{A}$ in $\mathcal{B}$, we can produce a structure $\mathcal{A}^{\mathcal{C}}$ such that if $\mathcal{C} \cong \mathcal{B}$, then $\mathcal{A}^{\mathcal{C}} \cong \mathcal{A}^{\mathcal{B}}$. We can then produce an index for $\mathcal{A}^{\mathcal{C}}$, and use the index set of $\mathcal{A}$ to check if it is isomorphic to $\mathcal{A}$. If it is not, then we know $\mathcal{C}$ is not isomorphic to $\mathcal{B}$. Otherwise, we need to check that the bi-interpretability does produce an isomorphism between $\mathcal{C}$ and $\mathcal{B}^{\mathcal{A}^{\mathcal{C}}}$, which $0''$ can check. One has to notice that all index sets compute $0''$, as their domain must be infinite, and all the formulas "$x_n = x_n$" must be in their atomic diagrams. (Given e, let $f(e)$ be a c.e. index so that $W_{f(e)} = D(\mathcal{A}) \setminus \{\ulcorner x_n = x_n \urcorner : n \notin W_e\}$. Then $f(e)$ is an index for $\mathcal{A}$ if and only if $W_e = \mathbb{N}$. So we get a reduction from $0''$ (TOT) to the index set of $\mathcal{A}$.) □

We will prove later that if $\mathcal{A}$ and $\mathcal{B}$ are effectively-bi-interpretable, then they also have the same Scott rank (Part 2) and their jumps are effectively-bi-interpretable too (see Remarks IX.0.13).

VI.2.2. Making structures into graphs. In this section, we show how every structure is effectively-bi-interpretable with a graph. This result will allow us to reduce statements about structures in general to statements about graphs, sometimes making proofs simpler. In Part 3, we will see that there are other classes of structures, like lattices or fields, that are also complete in the sense that every structure is effectively-bi-interpretable with one of them.

THEOREM VI.2.9. *For every structure $\mathcal{A}$, there is a graph $\mathcal{G}_{\mathcal{A}}$ that is effectively-bi-interpretable with $\mathcal{A}$. Furthermore, the interpretations are independent of the given structure. That is, given a vocabulary τ, the Σ^c_1 formulas used to define the sets involved in the interpretations are the same for all τ-structures $\mathcal{A}$.*

We only sketch the construction and let the reader verify the details.

The first step is to show that $\mathcal{A}$ is effectively-bi-interpretable with a structure $\mathcal{H}$ in the vocabulary $\{U, E\}$, where U is a unary relation and E a symmetric binary relation. The unary relation U picks out the elements that represent the domain of $\mathcal{A}$. The elements outside U are going to be used to code the relations in $\mathcal{A}$. Enumerate the domain of $\mathcal{A}$ as $\{a_0, a_1, ...\}$ and let $h_0, h_1, ...$ be the corresponding elements in $U^{\mathcal{H}}$. For each tuple $a_{i_1}, ..., a_{i_k}$ satisfying the nth relation R_n in τ (of arity k), we attach the following configuration to $h_{i_1}, ..., h_{i_k}$ in $\mathcal{H}$, where the top cycle has size $2n + 5$.

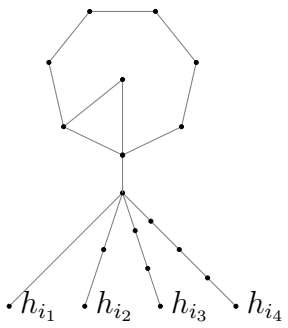


FIGURE 1. We call this configuration an *m-spider*, where m is the size of the top loop ($m = 7$ in this case). The edges represent the pairs of elements that satisfy E. Let ℓ_m be the number of nodes in the spider ($\ell_m = 13$ in this case).

So that both the interpretation of R and that of its complement are $\exists$-definable, to each tuple $a_{i_1}, ..., a_{i_k}$ not satisfying R_n we attach an $(2n + 4)$-spider.

It is clear that $\mathcal{A}$ can be effectively interpreted in $\mathcal{H}$: the domain of the interpretation is $U^{\mathcal{H}}$, and the interpretation of R_n is given by the set of tuples in $(U^{\mathcal{H}})^k$ that have a $(2n + 5)$-spider attached to them, which can be expressed by an $\exists$-formula. This set is also $\forall$-definable, as it is the set of tuples which do not have a $(2n + 4)$-spider attached to them.

Conversely, $\mathcal{H}$ can be interpreted in $\mathcal{A}$ as follows. Use A to interpret $U^{\mathcal{H}}$ and, for each m-spider attached to a tuple $h_{i_1}, ..., h_{i_k}$, use the elements

$$\langle\langle a_{i_1}, ..., a_{i_k}\rangle, m, 1\rangle, ..., \langle\langle a_{i_1}, ..., a_{i_k}\rangle, m, \ell_m\rangle \in \mathcal{A}^{<\mathbb{N}} \times \mathbb{N} \times \mathbb{N},$$

to interpret its elements. The domain of this interpretation is u.r.i. computable because, given a tuple of the form $\langle\langle a_{i_1}, ..., a_{i_k}\rangle, m, i\rangle$ with $i \leq \ell_m$, the tuple belongs to the interpretation if and only if $\langle a_{i_1}, ..., a_{i_k}\rangle \in R_n^{\mathcal{M}}$, where $n = \lfloor (m - 3)/2 \rfloor$. Similarly, we can also decide which pairs of these elements are E-connected.

Checking that the compositions of the interpretations are u.r.i. computable is also straightforward: the composition of the interpretations going from $\mathcal{A}$ to $\mathcal{H}$ and back is the identity; the interpretation going from $\mathcal{H}$ to $\mathcal{A}^{<\mathbb{N}}$ and back to $\mathcal{H}^{<\mathbb{N}}$ is a bit more tedious, but not much harder to analyze.

The second step is to show that every $\{\mathsf{U}, \mathsf{E}\}$-structure $\mathcal{H}$ is effectively-bi-interpretable with a graph $\mathcal{G} = (G; R)$. Within G, we will use a subset, G_0, to interpret the domain of $\mathcal{H}$. We use the other elements of G to encode the relations U and E on G_0. Enumerate the elements of H as $\{h_0, h_1, ...\}$ and the corresponding ones of G_0 as $g_0, g_1, ...$. Attach

to each element $g_i \in G_0$ one of the following two shapes, depending on whether $h_i \in U^{\mathcal{H}}$ or not.

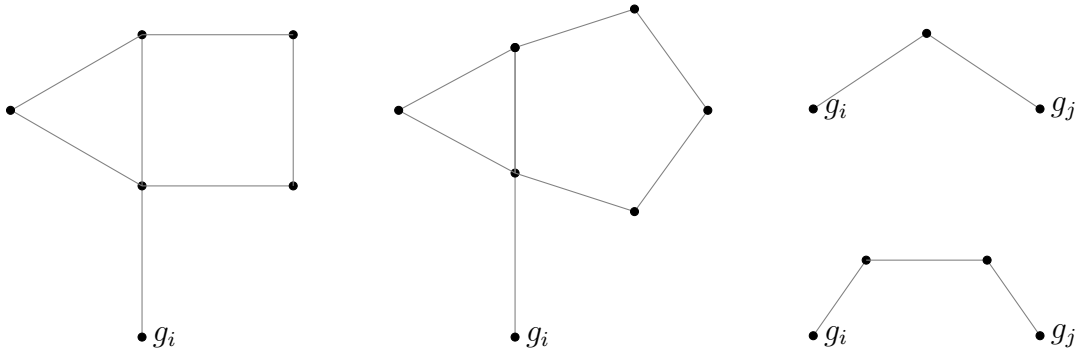


FIGURE 2. We call these configurations *4-flags*, *5-flags*, *2-connectors*, and *3-connectors*. We attach 3-flags to the elements that are in U and 4-flags to the ones out of U. We use 2-connectors to encode E.

Connect two elements of G_0 using a 2-connector if and only if the corresponding elements in $\mathcal{H}$ are connected by E. Connect them using a 3-connector if and only if the corresponding elements in $\mathcal{H}$ are not connected by E. The reason we cannot connect the elements of G_0 directly to code E is that we do not want to confuse the elements of G_0 with the ones used for the flags. This way, every element of G is either part of a flag (and hence out of G_0), attached to a flag (and hence in G_0), or attached to something that is attached to a flag (and hence part of either a 2-connector or a 3-connector, and out of G_0). Each of these three sets is $\exists$-definable, and hence G_0 is u.r.i. computable. Notice that the connectors coding the graph E among the elements of H do not get confused with these flags because since each edge in E is replaced by at least a 2-connectors, the smallest cycles one could produce are 6-cycles coming from triangles in $\mathcal{H}$.

The relation E is coded by the pairs of elements of G_0 which are connected by a 2-connector or, equivalently, not connected by a 3-connector. This is u.r.i. computable.

Again, checking that the composition of the interpretations are u.r.i. computable is straightforward.

Similar constructions can be found in [**HKSS02**].

CHAPTER VII

Finite-injury constructions

The technique of finite-injury constructions is among the most important ones in computability theory, and is used all throughout the field. It was introduced independently by Friedberg [**Fri57b**] and Muchnik[**Muc56**] to solve Post's problem, as we explain below. This technique is used to build computable objects using $0'$-computable information. On a computable construction, we can only only guess at this non-computable information, so we will be constantly taking steps in the wrong direction based on wrong guesses, and we will have to be able to recover from those mistakes.

We will see two kinds of finite-injury constructions. The first will be priority constructions, and the second true-stage constructions. Depending on the type of construction we want to do, one could be better than the other.

In a priority construction, one needs to build an object satisfying an infinite list of requirements whose actions are in conflict with one another — when requirements act, they may *injure* the work of other requirements. To control these injuries, requirements are listed in an order of priorities: Requirements are only allowed to injure weaker-priority requirements. In the type of constructions we will see, each requirement will be injured at most finitely many times, and hence there will be a point after which it is never injured again.

A true-stage construction works in quite a different way. It is based on a combinatorial device, the approximation of the true stages, which organizes our guesses on $0'$-computable information. The advantage of this combinatorial device is that it can be generalized to the iterates of the jump, even over the transfinite, as we will see in Part 2.

VII.1. Priority Constructions

To show how priority constructions work, we give a full proof of the Friedberg–Muchnik solution to Post's problem — a seminal result in computability theory. Post [**Pos44**] asked whether there was a computably enumerable set that was neither computable nor complete. That question was open for more than a decade, until Friedberg and Muchnik solved it independently by developing the method of priority constructions.

We will see two more finite-injury priority constructions in Chapter VIII on computable categoricity. We recommend that the reader interested in learning priority constructions read Theorem VIII.4.3 after fully understanding the proof below. The third finite-injury priority construction, Lemma VIII.5.8, is more complicated.

Theorem VII.1.1 (Friedberg [**Fri57b**], Muchnik[**Muc56**]). *There is a low, non-computable, computably enumerable set.*

Recall that a set A is *low* if $A' \equiv_T 0'$. Low sets cannot be complete.

PROOF. We build A as the union of a computable sequence of finite sets $A_0 \subseteq A_1 \subseteq A_2 \subseteq \cdots$ satisfying the following requirements for each $e \in \mathbb{N}$:

Negative requirements $\mathbf{N_e}$: If $\Phi_{e,s}^{A_s}(e) \downarrow$ for infinitely many s's, then $\Phi_e^A(e) \downarrow$.

Satisfying the N_e requirements for all $e \in \mathbb{N}$ ensures that A is low: We would get that $e \in A'$ if and only if $\Phi_{e,s}^{A_s}(e) \downarrow$ for infinitely many s's. This makes A' a Π_2^0 set. Since A' is already Σ_2^0, we get that A' is Δ_2^0.

Positive requirements $\mathbf{P_e}$: If W_e is infinite, then $A \cap W_e \neq \emptyset$.

Satisfying the P_e requirements for all $e \in \mathbb{N}$ ensures that the complement of A is not c.e., and hence that A is not computable — well, that is unless A is co-finite. We will also make sure during the construction that A is co-infinite.

We list these requirements by decreasing order of priority as follows:

$$N_0, P_0, N_1, P_1, N_2, P_2, ...,$$

the ones to the left having stronger priority than those to the right. Notice that each requirement has only finitely many requirements that are stronger than it. We think of each requirement as an individual worker trying to achieve its goal. Except for possible injuries, the different requirements will work almost independently of each other. Let us look at each of these requirements individually.

Negative requirements $\mathbf{N_e}$: The only way in which N_e would not be satisfied is if $\Phi_{e,s}^{A_s}(e)$ goes back and forth between converging and not converging infinitely often. Thus, what N_e needs to do is the following: If it sees that $\Phi_{e,s}^{A_s}(e)$ converges, it needs to try to preserve this computation forever by restraining elements to go into A below the use of this computation. Here is what N_e does at a stage s. Let r_e be the use of $\Phi_{e,s}^{A_s}(e) \downarrow$, that is, the length of the initial segment of the oracle A_s used in the computation $\Phi_{e,s}^{A_s}(e) \downarrow$. If the computation diverges, let $r_e = 0$. During the construction, N_e will not enumerate any number into A. Instead, it will impose a restraint on weaker-priority P_i requirements, not allowing them to enumerate elements below r_e into A. (This is why we call the N_i *negative* requirements.) N_e is not allowed to impose anything on stronger-priority requirements, which may enumerate elements below r_e and *injure* N_e.

Positive requirements $\mathbf{P_e}$: It is the P_e requirements that enumerate elements into A. (This is why we call them *positive* requirements.) They will enumerate at most one element each. The plan to satisfy P_e is to wait until we see some number enter W_e and enumerate it into A. However, we cannot enumerate just any number, as there are a couple things we need worry about. First, P_e is not allowed to injure stronger-priority requirements. In other words, if we let $R_e = \max_{i \leq e} r_i$, then P_e is not allowed to enumerate any number below R_e into A. Second, we want to make sure A is co-infinite. To do this, we only allow P_e to enumerate numbers that are greater than $2e$. The plan for P_e can now be restated as follows: At a stage $s > e$, if $W_{e,s} \cap A_s \neq \emptyset$, we consider P_e *done*, and we do not need to ever do anything else for P_e. Otherwise, if there is an $x \in W_{e,s}$ greater than $2e$ and greater than R_e, we say that P_e *requires attention*. Once P_e requires attention, it *acts* by enumerating such x into A.

The construction: Let us now describe the full construction. At stage 0, let $A_0 = \emptyset$. At each stage $s > 0$, do the following. First, define r_e for each $e < s$; recall that r_e is the use of $\Phi_{e,s}^{A_s}(e)$. Second, check which requirements P_e, for $e < s$, require

attention, and let them act; that is, for each $e < s$, if $W_{e,s} \cap A_s = \emptyset$ and there exists $x \in W_{e,s}$ with $x > \max(2e, R_e)$, add x to A_{s+1}. If no requirement requires attention, move on to the next stage without doing anything.

Verification: Each requirement P_e acts at most once. Therefore, a requirement N_e is injured at most $e-1$ times, and there is a stage after which it is never injured again. After this stage, if $\Phi_{e,s}^{A_s}(e)$ never converges again, N_e is satisfied. Otherwise, $\Phi_{e,t}^{A_t}(e)\downarrow$ for some later stage t. At stage t, N_e will define r_e to be the use of this computation. After t, no requirement of weaker priority is allowed to enumerate numbers below r_e. Since we are assuming all stronger-priority P_i requirements that ever act have acted already, we get that $A_t \upharpoonright r_e$ is preserved forever, and hence so is the computation $\Phi_{e,t}^{A_t}(e)\downarrow$, getting $\Phi_e^A(e)\downarrow$. N_e is then satisfied. In either case, r_e is eventually constant, either eventually equal to zero if $\Phi_e^A(e)\uparrow$, or eventually equal to the use of $\Phi_e^A(e)\downarrow$. Since this is true for all e, R_e is also eventually constant.

Let us now verify that the requirements P_e are all satisfied. If a requirement P_e ever requires attention, it acts, and then it is satisfied forever. Suppose that, otherwise, there is a stage t after which P_e never requires attention again. Assume t is large enough so that R_e has reached its limit already. Either P_e does not require attention because it is done, in which case we are done, or because all the numbers in W_e are below $\max(2e, R_e)$. In that case, P_e is trivially satisfied because W_e would be finite.

Finally, let us notice that A is co-infinite, as it can have at most e elements below $2e$ for each e. This is because only the requirements P_i for $i < e$ are allowed to enumerate numbers below $2e$. □

VII.2. The method of true stages

Often in computability theory, we want to use Δ^0_2 information to construct computable objects. We then need to computably approximate or guess the Δ^0_2 information. This can get messy, and there are various ways to organize this guessing system. We will concentrate on the method of *true stages for the enumeration of* $0'$, introduced by Lachlan in [**Lac73**]. There are slightly different definitions in the literature — we use our own, which is quite flexible and applies to a large variety of situations. The reason for our choice is that, in Part 2, we will be able to extend this notion throughout the hyperarithmetic hierarchy, obtaining a very powerful technique.

One way of approximating the halting problem $0'$ is by the sequence of finite sets

$$0'_s = \{e \in \mathbb{N} : \Phi_{e,s}(e)\downarrow\} \subseteq \mathbb{N}.$$

Notice $0'_s$ is finite. It is then natural to view $0'_s$ as a finite string, say by considering $0'_s \restriction m_s \in 2^{m_s+1}$, where $m_s = \max(0'_s)$. (Recall that $\mathcal{X} \restriction m$ is $\{x \leq m : x \in X\}$, or, when viewed as strings, it is the initial segment of X of length $m+1$.) The problem with this finite string is that it may always be wrong: It could be that at no stage $s > 0$ is $0'_s \restriction m_s$ an initial segment of $0'$, viewed as a sequence in $2^{\mathbb{N}}$. This might indeed be a problem on some constructions. Lachlan's idea was to consider $0'_s \restriction k_s$, where k_s is the least element enumerated into $0'$ at stage s (i.e., $k_s = \min(0'_s \setminus 0'_{s-1})$). The key difference is that there are infinitely many stages where $0'_s \restriction k_s$ is correct, in the sense that $0'_s \restriction k_s$ is an initial string of $0' \in 2^{\mathbb{N}}$. Stages where our guesses for $0'$ are correct are called *true stages*.

We introduce a different approximation to the jump that enjoys better combinatorial properties. Instead of $0'$, we will use the increasing settling time function for $0'$, which

we call ∇. At each stage s, we will computably define a finite string $\nabla_s \in \mathbb{N}^{<\mathbb{N}}$ which tries to approximate $\nabla \in \mathbb{N}^{\mathbb{N}}$. A true stage will be one where ∇_s is correct; i.e., it is an initial segment of ∇. One of the main advantages of using ∇ and ∇_s is that they relativize easily, allowing us to iterate them, as we will see in Part 2.

VII.2.1. The increasing settling time function. The settling time function of a c.e. set measures how fast its elements are enumerated. For now, we only consider the one for the halting problem. It has many uses in various constructions, and we will see a couple of examples in Subsection VII.2.2. We will deviate slightly from the standard settling time function to consider the strictly increasing version.

VII.2.1.1. *The definition of* ∇. The settling time function of a set measures the time a given enumeration takes to settle on an initial segment of the set.

Definition VII.2.1. The *i-th true stage (in the enumeration of $0'$)*, denoted $\nabla(i)$, is defined by recursion on i by any of the following three equivalent definitions:

$$\begin{aligned}
\nabla(i) &= \text{the least } t > \nabla(i-1) \text{ such that } 0'_t \restriction i = 0' \restriction i,\\
&= \text{the least } t > \nabla(i-1) \text{ such that } \Phi_i(i)\downarrow \iff \Phi_{i,t}(i)\downarrow,\\
&= \begin{cases} \nabla(i-1)+1 & \text{if } \Phi_i(i) \text{ diverges},\\ \nabla(i-1)+1 & \text{if } \Phi_i(i) \text{ converges by stage } \nabla(i-1),\\ \mu t(\Phi_{i,t}(i)\downarrow) & \text{if } \Phi_i(i) \text{ converges after stage } \nabla(i-1). \end{cases}
\end{aligned}$$

We use the value $\nabla(-1) = 1$ as the base case for the recursion, so that $\nabla(0) > 1$. We call t a *true stage* if $t = \nabla(i)$ for some i. We call ∇ the *increasing settling time function* for $0'$.

It is not hard to see that $\nabla \equiv_T 0'$: Clearly $\nabla \leq_T 0'$. To see the other direction, notice that $i \in 0' \iff i \in 0'_{\nabla(i)}$.

Lemma VII.2.2. *The set of true stages is co-c.e., and the set*

$$\{(i,t) \in \mathbb{N}^2 : t < \nabla(i)\}$$

is c.e.

Proof. Let us first observe that the set of initial segments of ∇, $\{\nabla \restriction i : i \in \mathbb{N}\} \subseteq \mathbb{N}^{<\mathbb{N}}$, is Π^0_1. To see this, note that $\sigma = \nabla \restriction i$ if and only if, for every $e < i$,

- either $\Phi_e(e)\uparrow$ and $\sigma(e) = \sigma(e-1)+1$,
- or $\Phi_{e,\sigma(e)}(e)\downarrow$ and $\sigma(e)$ is the least $t > \sigma(e-1)$ such that $\Phi_{e,t}(e)\downarrow$.

It follows that the set of true stages, i.e., the image of ∇, is Π^0_1: This is because t is a true stage if and only if there exists an increasing finite string σ whose last value is t (and the previous values are less than t) that is an initial segment of ∇.

As for the second part of the statement, $t < \nabla(i)$ if and only if there is no $\sigma \in (t+1)^{i+1}$ which is an initial segment of ∇. □

VII.2.1.2. *Domination properties.* One of the useful properties of ∇ is that it grows rapidly when compared to computable functions. For functions $f, g\colon \mathbb{N} \to \mathbb{N}$, we say that

- f *majorizes* g if $(\forall m)\ f(m) \geq g(m)$;
- f *dominates* g if $(\exists n)(\forall m \geq n)\ f(m) \geq g(m)$.

The function ∇ is fast growing in this sense: It dominates all computable functions (Lemma VII.2.5), and every function that dominates ∇ computes $0'$ (Lemma VII.2.3).

LEMMA VII.2.3. *If $g\colon \mathbb{N} \to \mathbb{N}$ dominates ∇, then g computes $0'$.*

PROOF. First, modify the first few values of g to get a function $f \equiv_T g$ that majorizes ∇. Given $x \in \mathbb{N}$, we can decide whether $x \in 0'$ by checking if $x \in 0'_{f(x)}$. □

COROLLARY VII.2.4. *Every subset of the set of true stages computes $0'$.*

PROOF. If we enumerate in increasing order the elements of a subset of the set of true stages, we obtain a function that dominates ∇. □

LEMMA VII.2.5. *Every computable function is dominated by ∇.*

PROOF. Let $g\colon \mathbb{N} \to \mathbb{N}$ be a computable function. We may assume g is non-decreasing, as otherwise we can always replace it with a larger non-decreasing computable function, for instance, $m \mapsto \max_{i \leq m} g(m)$.

We will find a sequence $\{s(i) : i \in \mathbb{N}\}$ for which $\nabla(s(i))$ is very large. More concretely, we need

$$\nabla(s(i)) \geq g(s(i+1)) \quad \text{for all } i.$$

This way, for every $m > s(0)$, if we let i be such that $s(i) < m \leq s(i+1)$, we have

$$g(m) \leq g(s(i+1)) \leq \nabla(s(i)) < \nabla(m).$$

Thus, for each $i \in \mathbb{N}$, we need to find a number $s(i)$ so that $\Phi_{s(i)}(s(i))$ takes at least $g(s(i+1))$ steps to converge. The difficulty is that, to define $s(i)$, we need to know the value of $s(i+1)$. This calls for the use of the — always rather mysterious — recursion theorem. For each e, we will define a computable function $s_e\colon \mathbb{N} \to \mathbb{N}$ uniformly in e, and we will let $s(i) = s_{e_0}(i)$, where e_0 is such that $s_{e_0} = \Phi_{e_0}$. Intuitively, you may think that while we are defining s as s_{e_0}, we already know e_0 is a computable index for s_{e_0}. The only reason this is not circular is that we must define s_e to be a total function, even if Φ_e is not — maybe it is still a bit circular, but it works.

We define s_e as follows: Let $s_e(i)$ be an index for the partial computable function that, on any input x, ignores x, waits for $\Phi_e(i+1)$ to converge, and then outputs $g(\Phi_e(i+1))$. That is,

$$\Phi_{s_e(i)}(x) = g(\Phi_e(i+1)).$$

Notice that s_e is total, even if $\Phi_e(i+1) \uparrow$, because $s_e(i)$ outputs just an index for this program, independent of whether the program computes a total function or not. We can also make s_e increasing by the Padding Lemma (see Chapter). By the recursion theorem (see Chapter), there is an e_0 such that $s_{e_0} = \Phi_{e_0}$. Let $s(i) = s_{e_0}(i)$. Then

$$\Phi_{s(i)}(x) = \Phi_{s_{e_0}(i)}(x) = g(\Phi_{e_0}(i+1)) = g(s_{e_0}(i+1)) = g(s(i+1)),$$

for every $x \in \mathbb{N}$, and in particular for $x = s(i)$. Since the outcome of $\Phi_{s(i)}(s(i))$ is $g(s(i+1))$, it takes at least $g(s(i+1))$ steps to converge (as that is what it takes to just write that number down in the output tape). Thus, $\nabla(s(i)) \geq g(s(i+1))$ as wanted. □

VII.2.2. A couple of examples. Just the fact that the set of true stages is co-c.e., and that ∇ grows so fast, are enough to make ∇ useful in computability theory and computable structure theory. We give a couple of examples to illustrate its use. In Section I.1.2, we built a copy $\mathcal{A}$ of the ordering $\omega = (\mathbb{N}; \leq)$ so that the isomorphism between $\mathcal{A}$ and $(\mathbb{N}; \leq)$ computes $0'$. We now produce another such copy $\mathcal{A}$ using a different method.

LEMMA VII.2.6. *There is a computable ω-presentation $\mathcal{A}$ of the ordering $(\mathbb{N}; \leq)$ such that any embedding from $\mathcal{A}$ to $(\mathbb{N}; \leq)$ computes $0'$.*

PROOF. The idea is to define $\mathcal{A} = (A; \leq_A)$ together with a computable sequence $a_0 <_A a_1 <_A a_2 <_A \cdots$ such that there are at least $\nabla(i)$ elements $<_A$-below a_{i+1} for every i. This way, if $g\colon A \to \mathbb{N}$ is an embedding from $\mathcal{A}$ to $(\mathbb{N}; \leq)$, we would have that the function $i \mapsto g(a_{i+1})$ majorizes ∇ and hence computes $0'$. Recall that the set $\{(i,t) : t < \nabla(i), i \in \mathbb{N}\}$ is c.e. We build $\mathcal{A}$ by first laying down elements $a_0 <_A a_1 <_A a_2 <_A \cdots$ (say, using the even integers: $a_n = 2n$), and then adding elements $b_{i,t} \leq_A$-in-between a_i and a_{i+1} for each i, t with $t < \nabla(i)$. More formally, if f is a computable one-to-one enumeration of $\{(i,t) : t < \nabla(i), i \in \mathbb{N}\}$, name the odd number $2n+1$ with the label $b_{i,t}$ if $f(n) = (i,t)$ and then define

$$a_j <_A b_{i,t} \iff j \leq i \quad \text{and} \quad b_{j,s} <_A b_{i,t} \iff j < i \vee (j = i \wedge s < t).$$

We then get that there are $\nabla(i)$ elements $\leq_A$-below a_{i+1} as needed. □

The following lemma answers the question of how difficult is it to find a basis on a vector space. A jump is sufficient, as we can computably enumerate a maximal linearly independent set using the linear dependence relation, which we know is r.i.c.e.The lemma below shows it is necessary.

LEMMA VII.2.7. *There is a computable copy of the infinite dimensional $\mathbb{Q}$-vector space $\mathbb{Q}^\infty$ where every basis computes $0'$.*

We will actually show that every infinite linearly independent set in this ω-presentation computes $0'$. Let $\mathbb{Q}^\infty$ denote the standard ω-presentation of the infinite dimensional $\mathbb{Q}$-vector space, which has a computable basis $\{e_i : i \in \mathbb{N}\}$.

PROOF. The idea is to define a copy $\mathcal{A}$ of $\mathbb{Q}^\infty$ by taking the quotient of $\mathbb{Q}^\infty$ over a computable subspace U with infinite co-dimension. The equivalence relation generated by a computable subspace U — namely $u \sim v \iff u - v \in U$ — is computable, and hence we have a computable congruence presentation $\mathcal{A}$, where the projection map from $\mathbb{Q}^\infty$ to $\mathcal{A}$ is also computable (see Lemma I.1.9).

Define U so that, for every s_1 and s_2 which are not true stages, e_{s_1} and e_{s_2} are linearly dependent in $\mathbb{Q}^\infty/U$. To get this, all we need to do is add to U a vector of the form $ae_{s_1} - e_{s_2}$ for some $a \in \mathbb{Q}$, as soon as we realize s_1 and s_2 are not true. Before showing how to define U in a computable way, let us see why having such a U is enough. Suppose $I \subseteq A$ is an infinite linearly independent set in $\mathcal{A}$; we need to show $I \geq_T 0'$. Since the projection map is computable, we can get an infinite set $J \subseteq \mathbb{Q}^\infty$ which is not just linearly independent, but also linearly independent modulo U. The subspace generated by $e_0, e_1, ..., e_{\nabla(n)-1}$ has dimension $n+1$ when projected to $\mathcal{A}$, because, except for $e_{\nabla(0)}, ..., e_{\nabla(n-1)}$, all the other vectors are linearly dependent

among themselves. Therefore, if we take $n+2$ vectors $v_0, ..., v_{n+1}$ from J, they cannot all belong to the subspace of $\mathbb{Q}^\infty$ generated by $e_0, e_1, ..., e_{\nabla(n)-1}$. Recall that in $\mathbb{Q}^\infty$, every vector is given as a linear combination of the bases of e_i's. One of the vectors v_i for $i \leq n+1$ must then use some e_t for $t \geq \nabla(n) - 1$ in its representation. Let $g(n)$ be the largest t such that e_t appears in the representation of one of the vectors v_i for $i \leq n+1$. The function g majorizes ∇, and hence we can use g to compute $0'$ as in Lemma VII.2.3.

We now have to show how to build U effectively. At each stage s, we define a subset U_s of the set

$$V_s = \{\sum_{i<s} \frac{p_i}{q_i} e_i : p_i, q_i \in \mathbb{Z}, |p_i| < s, 0 < q_i < s\},$$

and, at the end, define $U = \bigcup_{s\in\mathbb{N}} U_s$. If we had that $U \cap V_s = U_s$ for every s, then U would be computable. Therefore, after each stage s, we must take care that no element of $V_s \setminus U_s$ later enters U. To get U to be a subspace, we need to have each U_s closed under linear combinations within V_s.

Suppose that, at stage s, we discover that s_1 and s_2 are not true stages and we have not made e_{s_1} and e_{s_2} independent in $\mathcal{A}$ yet. (Recall that the set of non-true stages is c.e.) We then want to add a vector of the form $ae_{s_1} - e_{s_2}$ to U so that we make e_{s_1} and e_{s_2} dependent in $\mathcal{A}$ without changing U within V_s: All we have to do is search for such an $a \in \mathbb{Q}$ such that when we add $ae_{s_1} - e_{s_2}$ to U, we keep all the vectors in $V_s \setminus U_s$ outside U. That is, we need to make sure that no vector in $V_s \setminus U_s$ belongs to the subspace generated by $U_s \cup \{ae_{s_1} - e_{s_2}\}$. We need to show at least one such a exists. Using basic linear algebra, if $a_0 \neq a_1$, and e_{s_1} and e_{s_2} are independent over U_s, then the intersection of the spaces generated by $U_s \cup \{a_0 e_{s_1} - e_{s_2}\}$ and by $U_s \cup \{a_1 e_{s_1} - e_{s_2}\}$ is the subspace generated by U_s. Since V_s is finite, there can be at most finitely many a's which generate elements in $V_s \setminus U_s$. In other words, for all but finitely many a's, the space generated by $U_s \cup \{ae_{s_1} - e_{s_2}\}$ adds no new vectors to V_s that were not in the subspace generated by U_s already. Now that we know such a exist, all we have to do is look for one.

Notice that U has infinite co-dimension as whenever $t_1, ..., t_k$ are true stages, $e_{t_1}, ..., e_{t_k}$ are linearly independent modulo U, as no vector in the subspace generated by them was ever added to U. □

VII.3. Approximating the settling time function

Every true stage can figure out all the previous true stages in a uniformly computable way. More precisely: Suppose $t = \nabla(i)$ is the ith true stage. Using the fact that $0'_t \restriction i = 0' \restriction i$, we have that, for $j \leq i$, $\nabla(j)$ is the least $s > \nabla(j-1)$ such that $0'_s \restriction j = 0'_t \restriction j$. If t is not a true stage, we can still apply the same procedure and get the stages $s < t$ that t *believes* should be true.

Definition VII.3.1. Given $j < t$, we define the *jth apparent true stage at t*, denoted $\nabla_t(j)$, as the least $s \leq t$ such that $s > \nabla_t(j-1)$ and $0'_s \restriction j = 0'_t \restriction j$. Again, to match with ∇, we are using $\nabla_t(-1) = 1$ in the definition of $\nabla_t(0)$.

This definition only makes sense if $s \leq t$, so once we reach a j with $\nabla_t(j) = t$, we cannot define any more apparent true stages, and we let ∇_t be the string defined up to that point. Thus, ∇_t is a finite increasing string whose last element is always t.

From the paragraph preceding the definition, we get that if t is the ith true stage, then $\nabla_t = \nabla \restriction i$. Furthermore, for every $s > t$, since $0'_t \restriction i = 0'_s \restriction i = 0' \restriction i$, we get that $\nabla_s \restriction i$ is also correct and equal to $\nabla \restriction i$. On the other hand, if t is not a true stage, since t is the last entry of ∇_t, we have that $\nabla_t \not\subseteq \nabla$. For the same reason, if $s > t$ is a true stage, then $\nabla_t \not\subseteq \nabla_s$. In short, for $t \in \mathbb{N}$,

$$t \text{ is a true stage} \iff \nabla_t \subset \nabla \iff \forall s > t\ (\nabla_t \subseteq \nabla_s).$$

By an argument similar to the above, we get the following property:

(♣) For every $r < s < t$, if $\nabla_r \subseteq \nabla_t$, then $\nabla_r \subseteq \nabla_s$.

The reason is that if $\nabla_r \subseteq \nabla_t$, then no number below $|\nabla_r|$ is enumerated into $0'$ between the stages r and t. That would then also be true between the stages r and s, and hence $\nabla_r \subseteq \nabla_s$.

The following two lemmas are intended to give us a feeling for how the sequence $\{\nabla_s : s \in \mathbb{N}\}$ behaves. Let $\mathcal{T}$ be the image of the function $s \mapsto \nabla_s$. We recommend the reader try to draw a picture of $\mathcal{T}$, and see how the sequence $\{\nabla_s : s \in \mathbb{N}\}$ moves around $\mathcal{T}$.

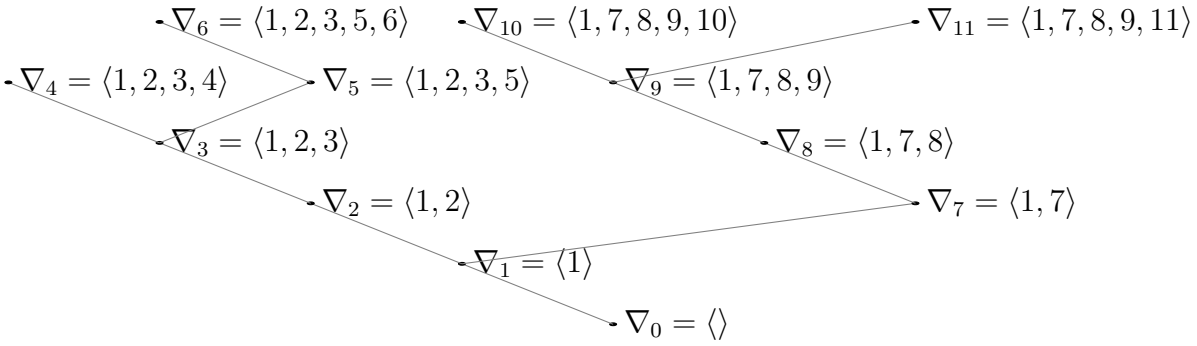


FIGURE 1. Example where 3 is enumerated into $0'$ at stage 5, 1 at stage 7 and 4 at stage 11.

LEMMA VII.3.2. *The set $\mathcal{T} = \{\nabla_s : s \in \mathbb{N}\} \subseteq \mathbb{N}^{<\mathbb{N}}$ is a computable tree whose only path is ∇.*

PROOF. $\mathcal{T}$ is computable because given $\sigma \in \mathbb{N}^{<\mathbb{N}}$, we can calculate ∇_t, where t is the last entry of σ, and then check if $\sigma = \nabla_t$.

To show that $\mathcal{T}$ is a tree, we need to show that it is closed downward. To do this, all we have to observe is that if $\nabla_s(i) = t$, then $\nabla_s \restriction i = \nabla_t$. This is because $0'_s \restriction i = 0'_t \restriction i$, and hence the computations of $\nabla_t \restriction i$ and $\nabla_s \restriction i$ are the same.

About the paths of $\mathcal{T}$, clearly ∇ is one of them. We claim that if $\nabla_s \not\subset \nabla$, the set of extensions of ∇_s in $\mathcal{T}$ is finite, and hence there is no path extending ∇_s. Let $t > s$ be a true stage. Then $\nabla_s \not\subseteq \nabla_t$. By (♣), for all $u \geq t$, $\nabla_s \not\subseteq \nabla_u$. □

LEMMA VII.3.3. *The Kleene-Brower ordering, $\leq_{KB}$, on $\mathcal{T}$ produces a computable ordering of order type $\omega + \omega^*$ on which every descending sequence computes $0'$.*

The *Kleene-Brower ordering*, $\leq_{KB}$, on $\mathbb{N}^{<\mathbb{N}}$ is defined as follows: $\sigma \leq_{KB} \tau$ if either σ and τ are incomparable and, for the least i with $\sigma(i) \neq \tau(i)$, we have $\sigma(i) < \tau(i)$, or σ and τ are comparable as strings and $\sigma \supseteq \tau$.

PROOF. To prove that $(\mathcal{T}; \leq_{KB}) \cong \omega + \omega^*$, we prove that if s is a true stage, then there are only finitely many strings in $\mathcal{T}$ that are $\geq_{KB} \nabla_s$; and if s is not a true stage, then there are only finitely many strings in $\mathcal{T}$ that are $\leq_{KB} \nabla_s$. For the former claim, if s is a true stage, then for every $t \geq s$, we have $\nabla_t \supseteq \nabla_s$, and hence $\nabla_t \leq_{KB} \nabla_s$. For the latter claim, if s is not a true stage and $t > s$ is a true stage, then there is a least i such that $\nabla_s(i) \neq \nabla_t(i)$. The reason for this difference must be that $i \notin 0'_s$ while $i \in 0'_t$, and hence $\nabla_t(i) > s \geq \nabla_s(i)$. Since t is true, we have that, for every $u \geq t$, $\nabla_u \supseteq \nabla_t$, and hence $\nabla_u \restriction i = \nabla_s \restriction i$ and $\nabla_u(i) = \nabla_t(i) > \nabla_s(i)$. Thus, $\nabla_u \geq_{KB} \nabla_s$.

Every descending sequence must be a subsequence of $\{\nabla_t : t \text{ is a true stage}\}$, and hence computes $0'$ by Corollary VII.2.4. □

EXERCISE VII.3.4. Show that $(\mathcal{T}; \leq_{KB})$ has a computable ascending sequence.

EXERCISE VII.3.5. (Hard) Use a priority argument to show that there is an ω-presentation of $\omega + \omega^*$ which has no computable ascending sequence and no computable descending sequence.

REMARK VII.3.6. Hirschfeldt and Shore [**HS07**, Theorem 2.11] showed that every ω-presentationof $\omega + \omega^*$ must have either an ascending sequence or a descending sequence that is low relative to the ω-presentation.

EXERCISE VII.3.7. A small modification of the proof of Theorem V.4.3 can produce another interesting spectrum. We view a set $\Gamma \subseteq 2^{<\mathbb{N}}$ as a c.e operator by letting $\Gamma^X = \{|\tau| : \tau \subseteq X, \tau \in \Gamma\}$. Given a finite set $F \subseteq \mathbb{N}$, let

$$\Gamma_F = \{\nabla \restriction i : i \in F\} \cup \{\tau \in 2^{<\mathbb{N}} : \tau \not\subset \nabla\}.$$

Notice that $\Gamma_F^{\nabla} = F$. Consider the family of sets:

$$\mathcal{F} = \{\Gamma_F \oplus \{n\} : F \subseteq \mathbb{N} \text{ finite } \& \ F \neq W_n^{\nabla}\}.$$

Prove that

$$DgSp(\mathcal{G}_{\mathcal{F}}) = \{X \in 2^{\mathbb{N}} : X \text{ not } \Delta^0_2\}.$$

(The first one to construct a structure with this spectrum was Kalimullin [**Kal08**]. The construction above is due to Montalbán [**ACK**$^+$, Theorem 2].)

EXERCISE VII.3.8. Define a c.e. set A as follows: At stage s, if $W_{e,s} \cap A_s = \emptyset$ and $(\exists x \in W_{e,s})\ x > 2 \cdot \nabla_s(\nabla_s(e))$, enumerate x into A_{s+1}. Prove that A is low and non-computable.

VII.4. A construction of linear orderings

In this section, we prove a well-known result that is best proved using the method of true stages we just developed. Given linear orderings $\mathcal{A}$ and $\mathcal{B}$, we let $\mathcal{A} \cdot \mathcal{B}$ be the ordering on $A \times B$ given by $\langle a_0, b_0\rangle \leq_{\mathcal{A}\cdot\mathcal{B}} \langle a_1, b_1\rangle$ if either $b_0 \leq_{\mathcal{B}} b_1$, or $b_0 = b_1$ and $a_0 \leq_{\mathcal{A}} a_1$. Notice that the coordinates are compared from right to left, not as in the lexicographic ordering. It is just traditional notation. Then, for instance $\mathcal{A} + \mathcal{A} = \mathcal{A} \cdot 2$, and $\mathbb{Z} \cdot \mathcal{A}$ is the linear ordering obtained by replacing each element in $\mathcal{A}$ by a copy of $\mathbb{Z}$.

THEOREM VII.4.1. *Let $\mathcal{L}$ be a linear ordering. Then $\mathbb{Z} \cdot \mathcal{L}$ has a computable copy if and only if $\mathcal{L}$ has a $0''$-computable copy.*

The left-to-right direction is the easy one. On a computable copy of $\mathbb{Z} \cdot \mathcal{L}$, the equivalence relation $\sim$, given by $a \sim b$ if and only if they are finitely apart, is $0''$ computable, and hence we can make the copy of $\mathbb{Z} \cdot \mathcal{L}$ into a $0''$-computable congruence ω-presentation of $\mathcal{L}$.

The proof of the other direction is divided into a few steps which we prove in separate lemmas. The first lemma is a general one that will be useful in other settings too. It gives a way of approximating $0'$-computable structures in a way that correct approximations to the structure happen at the same stages where we have correct approximations to ∇.

LEMMA VII.4.2. *Let $\mathcal{B}$ be a $0'$-computable ω-presentation of a structure in a relational vocabulary τ. There is a computable sequence of finite $\tau_{|\cdot|}$-structures $\{\mathcal{B}_s : s \in \mathbb{N}\}$ such that*

$$(\forall s < t) \quad \nabla_s \subseteq \nabla_t \quad \Rightarrow \quad \mathcal{B}_s \text{ is a substructure of } \mathcal{B}_t,$$

and

$$\mathcal{B} = \bigcup \{\mathcal{B}_s : s \text{ a true stage}\}.$$

Moreover, if φ is a $\forall$-formula true of $\mathcal{B}$, we can make the $\mathcal{B}_s$'s satisfy φ too.

PROOF. Let $\mathcal{A}_t$ be the τ_t-substructure of $\mathcal{B}$ with domain $\{0, ..., t\}$. The sequence $\{\mathcal{A}_t : t \in \mathbb{N}\}$ is $0'$ computable. Let Φ be a computable function such that $\Phi^{\nabla}(t)$ is an index for the finite structure $\mathcal{A}_t$. If at a stage s we believe ∇_s is an initial segment of ∇, we also believe that Φ^{∇_s} outputs the indices of the first few structures in the sequence $\{A_t : t \in \mathbb{N}\}$. For each s, let t_s be the largest t so that, for every $i \leq t$, $\Phi^{\nabla_s}(i)$ converges and outputs an index for a finite structure $\tilde{\mathcal{A}}_i$ satisfying φ and so that

$$\tilde{\mathcal{A}}_0 \subseteq \tilde{\mathcal{A}}_1 \subseteq \cdots \subseteq \tilde{\mathcal{A}}_t.$$

Let $\mathcal{B}_s = \tilde{\mathcal{A}}_{t_s}$. We then have that if $\nabla_s \subseteq \nabla_r$, $\Phi^{\nabla_s}(i) = \Phi^{\nabla_r}(i)$ for all $i \leq t_s$, and hence $\mathcal{B}_s \subseteq \mathcal{B}_r$. If $\nabla_s \subseteq \nabla$, then $\tilde{\mathcal{A}}_{t_s}$ is actually one of the $\mathcal{A}_t$'s, and hence $\mathcal{B}_s \subset \mathcal{B}$. □

LEMMA VII.4.3. *If $\mathcal{L}$ has a $0'$-computable copy, then $(\mathbb{Z} \times \mathcal{L}; <, Adj)$ has a computable copy.*

PROOF. Let $\{\mathcal{L}_s : s \in \mathbb{N}\}$ be a sequence of finite linear orderings approximating $\mathcal{L}$ as in Lemma VII.4.2.

At each stage s, we build a finite linear ordering $\mathcal{A}_s = (\{0, ..., k_s\}; \leq_{A_s}, \mathrm{Adj}_s)$ and an onto, order-preserving map $g_s \colon \mathcal{A}_s \to \mathcal{L}_s$ such that $g_s(a) = g_s(b)$ if and only if there is a finite sequence of Adj_s-adjacent elements in between a and b in $\mathcal{A}_s$. The binary relations Adj_s satisfy that if $\mathcal{A}_s \models \mathrm{Adj}_s(a, b)$, then there is no element in between a and b in $\mathcal{A}_s$, but there could be elements $a, b \in \mathcal{A}_s$ without elements in between for which Adj_s does not hold. Thus, $\mathcal{A}_s$ is partitioned into *adjacency chains*, where an adjacency chain is a string of elements $a_0 <_{A_s} \cdots <_{A_s} a_k$ with $\mathrm{Adj}_s(a_i, a_{i+1})$ for all $i < k$ and such that for no b do we have $\mathrm{Adj}_s(b, a_0)$ or $\mathrm{Adj}_s(a_k, b)$. The condition on g above implies that for each $\ell \in \mathcal{L}_s$, $g_s^{-1}(\ell)$ is an adjacency chain.

At each stage s, we need to satisfy the following two properties:

(1) If $t \leq s$, then $\mathcal{A}_t \subseteq \mathcal{A}_s$.
(2) If $\nabla_t \subset \nabla_s$, then $g_t \subseteq g_s$, and for every $\ell \in \mathcal{L}_t$, $1 + g_t^{-1}(\ell) + 1 \subseteq g_s^{-1}(\ell)$.

Let us first note that these conditions are enough to build the desired structure $\mathcal{A}$. Condition (1) allows us to define a computable linear ordering with adjacencies

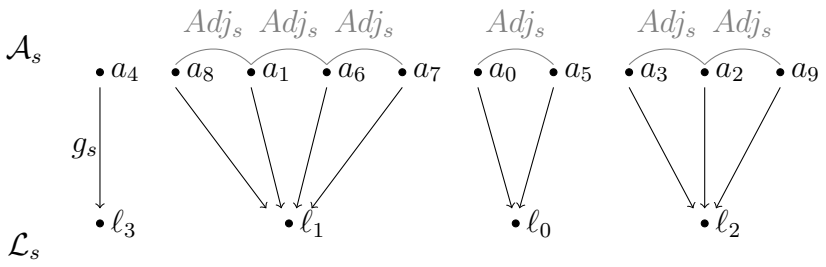


FIGURE 2. The top row are the points in $\mathcal{A}_s$ ordered by $\leq_{A_s}$ from left to right. The bottom row are the points in $\mathcal{L}_s$ ordered by $\leq_{L_s}$ from left to right.

$\mathcal{A} = \bigcup_s \mathcal{A}_s$. Condition (2) allows us to define an onto, order-preserving map $g = \bigcup\{g_s : s \text{ a true stage}\} \colon \mathcal{A} \to \mathcal{L}$. Furthermore, for every $\ell \in \mathcal{L}$, $g^{-1}(\ell)$ must be infinite and satisfy that any two elements in it are linked by a finite sequence of adjacencies. Therefore, $g^{-1}(\ell)$ is isomorphic to $\mathbb{Z}$, and we get that $\mathcal{A}$ is isomorphic to $\mathbb{Z} \cdot \mathcal{L}$.

Last, we need to show that, at each stage $s+1$, we can define $\mathcal{A}_{s+1}$ and g_{s+1} so they satisfy (1) and (2). Let $t \leq s$ be the largest such that $\nabla_t \subseteq \nabla_{s+1}$. Thus, we know that $\mathcal{L}_t \subseteq \mathcal{L}_{s+1}$, and we need to define $\mathcal{A}_{s+1}$ extending $\mathcal{A}_s$ and g_{s+1} extending g_t. The rest of the proof is just a brute-force combinatorial argument proving that such an $\mathcal{A}_{s+1}$ and g_{s+1} exist. We recommend the reader try to prove it before reading it, to understand the little intricacies of the proof.

First, define $\tilde{A}_{s+1}$ by adding a new element at the end of each adjacency chain in A_s, and by attaching each adjacency chain that was not in $\mathcal{A}_t$ to one that was. (To *attach* two adjacency chains, we add a new element in between the chains and make it satisfy Adj_{s+1} with the ends of the chains.) Extend $g_t \colon \mathcal{A}_t \to \mathcal{L}_t$ to $\tilde{g}_{s+1} \colon \tilde{A}_{s+1} \to \mathcal{L}_t$ so that, for each $\ell \in \mathcal{L}_t$, $\tilde{g}_{s+1}(\ell)$ is an adjacency chain in $\tilde{A}_{s+1}$.

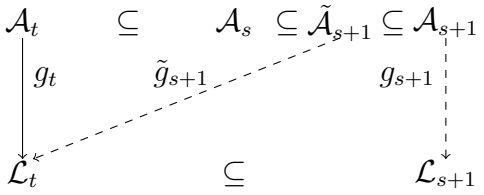


FIGURE 3. The diagram above commutes.

Second, define $A_{s+1} \supseteq \tilde{A}_{s+1}$ by adding a new element a_ℓ in between chains for each new $\ell \in \mathcal{L}_{s+1} \setminus \mathcal{L}_t$. Of course, if $\ell_0 < \ell < \ell_1$ with $\ell_0, \ell_1 \in \mathcal{L}_t$, then the a_ℓ must be in between the chains corresponding to $g^{-1}(\ell_0)$ and $g^{-1}(\ell_1)$. Finally, extend $\tilde{g}_{s+1} \colon \tilde{A}_{s+1} \to \mathcal{L}_t$ to $g_{s+1} \colon A_{s+1} \to \mathcal{L}_{s+1}$ by mapping each a_ℓ to ℓ. □

LEMMA VII.4.4. *If $(\mathbb{Z} \cdot \mathcal{L}; \leq, Adj)$ has a $0'$ computable copy, then $\mathbb{Z} \cdot \mathcal{L}$ has a computable copy.*

PROOF. Let $\mathcal{B}$ be the $0'$-computable copy of $(\mathbb{Z} \cdot \mathcal{L}; \leq, \mathrm{Adj})$. Let $\{\mathcal{B}_s : s \in \mathbb{N}\}$ be a sequence of finite structures approximating $\mathcal{B}$ as in Lemma VII.4.2. We assume each $\mathcal{B}_s$

satisfies the $\forall$-sentence saying that they are linear orderings and that if $\mathcal{B}_s \models \mathrm{Adj}(a,b)$, there is no element in between a and b. However, as for the structures $\mathcal{A}_s$ in the previous lemma, there will be elements a and b not satisfying $\mathrm{Adj}(a,b)$ in $\mathcal{B}_s$ and without anything in $\mathcal{B}_s$ in between them.

At each stage s, we build a finite linear ordering

$$\mathcal{A}_s = (\{0, ..., k_s\}; \leq_{A_s}, \mathrm{Adj}_s)$$

and an order-preserving, one-to-one map $h_s \colon \mathcal{B}_s \to \mathcal{A}_s$. Again, as with the structures $\mathcal{A}_s$ from the previous lemma, Adj_s satisfies $\forall a, b \leq k_s(\mathrm{Adj}_s(a,b) \wedge a <_{A_s} b \to \nexists c(a <_{A_s} c <_{A_s} b))$, and hence $\mathcal{A}_s$ is partitioned into *adjacency chains*. We do not require h_s to be onto, not even in the limit. Instead, all we require is that every adjacency chain in $\mathcal{A}_s$ has an element in the image of h_s. Also, we require that two elements of $\mathcal{B}_s$ are in the same adjacency chain if and only if their images are. Notice that we do not require h_s to preserve Adj, but only to preserve the property of being in the same adjacency chain.

At each stage s, we need to satisfy the following two properties:

(1) If $t \leq s$, then $(\{0, ..., k_t\}; \leq_{A_t}) \subseteq (\{0, ..., k_s\}; \leq_{A_s})$.
(2) If $\nabla_t \subseteq \nabla_s$, then $\mathcal{A}_t \subseteq \mathcal{A}_s$ and $h_t \subseteq h_s$.

Condition (1) allows us to define a computable linear ordering

$$\mathcal{A} = (\mathbb{N}; \leq_A) = \bigcup_s (\{0, ..., k_s\}; \leq_{A_s}).$$

Notice that we lose the adjacency relation. Condition (2) allows us to define an embedding $h = \bigcup\{h_s : s \text{ a true stage}\} \colon \mathcal{B} \to \mathcal{A}$, which preserves ordering and adjacencies. The embedding h produces a bijection between the adjacency chains in $\mathcal{B}$ and those in $\mathcal{A}$, and an embedding of each adjacency chain in $\mathcal{B}$ to the corresponding one in $\mathcal{A}$. Since the adjacency chains in $\mathcal{B}$ are isomorphic to $\mathbb{Z}$, the ones in $\mathcal{A}$ must also be isomorphic to $\mathbb{Z}$, and we get that $\mathcal{A}$ and $\mathcal{B}$ are isomorphic.

Last, we need to show that, at each stage $s+1$, we can define $\mathcal{A}_{s+1}$ and h_{s+1} so they satisfy (1) and (2). Let $t \leq s$ be the largest such that $\nabla_t \subseteq \nabla_{s+1}$. We need to define $(A_{s+1}; \leq_{s+1})$ extending $(A_s; \leq_s)$ and Adj_{s+1} and h_{s+1} extending Adj_t and h_t. The rest of the proof is just a brute-force combinatorial argument proving that such an $\mathcal{A}_{s+1}$, Adj_{s+1}, and g_{h+1} exist. Again, we recommend the reader try to prove it before reading it, to understand the little intricacies of the proof.

Define $\tilde{\mathrm{Adj}}_{s+1}$ on A_s so that it is compatible with Adj_t (ignoring Adj_s) and so that every element belongs to an adjacency chain that existed in $\mathcal{A}_t$. Extend $(A_s; \leq_s)$ to $(\tilde{A}_{s+1}; \leq_{s+1})$ by adding one new element a_ℓ for each $\ell \in B_{s+1} \setminus B_t$ so that we can extend $h_t \colon \mathcal{B}_t \to \mathcal{A}_t$ to $h_{s+1} \colon \mathcal{B}_{s+1} \to A_{s+1}$ (recall that $\mathcal{B}_t \subseteq \mathcal{B}_{s+1}$). Also, if two adjacency chains in $\mathcal{B}_t$ have collapsed to one in $\mathcal{B}_{s+1}$, we need to collapse the respective chains in $\mathcal{A}_{s+1}$: Thus, if two consecutive elements $\ell_0, \ell_1 \in \mathcal{B}_{s+1}$ belong to adjacency chains that were not separate chains in $\mathcal{B}_t$, but are part of a single chain in $\mathcal{B}_{s+1}$, we add a new element a_{ℓ_0,ℓ_1} to A_{s+1} in between the adjacency chains corresponding to $h_t(\ell_0)$ and $h_t(\ell_1)$ so that we can attach those chains. Define Adj_{s+1} on A_{s+1} so that h_{s+1} produces a bijection between the adjacency chains in $\mathcal{B}_{s+1}$ and those in $\mathcal{A}_{s+1}$. $\square$

EXERCISE VII.4.5. (Downey [**DK92**]) Prove that $\mathcal{L}$ has a $0'$ computable copy if and only if $(\mathbb{Q} + 2 + \mathbb{Q}) \cdot \mathcal{L}$ has a computable copy.

CHAPTER VIII

Computable Categoricity

Computably categorical structures are the ones for which all computable ω-presentations have the same computational properties. This is a desirable property, of course, but the structures which have it are rather few. The notion was originally introduced by Mal'cev [**Mal62**] in 1962 for groups, and has been intensively studied over the past few decades.

VIII.1. The basics

Most of the properties one considers in computable structure theory are invariant under computable isomorphisms, but not necessarily under all isomorphisms: Computable ω-presentations may be isomorphic and still have different computational properties. For instance, there are computable ω-presentations of the countable, infinite-dimensional $\mathbb{Q}$-vector space $\mathbb{Q}^\infty$ where all the finite-dimensional subspaces are computable, and there are computable ω-presentations of $\mathbb{Q}^\infty$ where no non-trivial finite-dimensional subspace is computable (see [**DHK+07**]).

Definition VIII.1.1. A computable structure $\mathcal{A}$ is *computably categorical* if there is a computable isomorphism between any two computable copies of $\mathcal{A}$.

The following somewhat trivial lemma shows how computably categorical structures are exactly the ones that avoid the behavior of the example above, that is, the ones where all computable copies have the same computable relations.

Lemma VIII.1.2. *Let $\mathcal{A}$ be a computable structure. The following are equivalent:*

(1) *$\mathcal{A}$ is computably categorical.*
(2) *For every $R \subseteq A^n$ and every computable copy $\mathcal{B}$ of $\mathcal{A}$, if R is computable, there is a computable $R^{\mathcal{B}} \subseteq B^n$ with $(\mathcal{B}, R^{\mathcal{B}}) \cong (\mathcal{A}, R)$.*

Proof. To show that (1) implies (2), consider a computable isomorphism $g\colon \mathcal{B} \to \mathcal{A}$, and define $R^{\mathcal{B}} = g^{-1}(R)$. For the other direction, consider a computable copy $\mathcal{B}$ of $\mathcal{A}$; we need to build a computable isomorphism between them. Of course, we are assuming $\mathcal{A}$ is infinite, and hence we may assume its domain is $\mathbb{N}$. Let

$$R = \{(n, n+1) : n \in \mathbb{N}\} \subseteq \mathbb{N}^2 = A^2.$$

Since R is computable, there is a computable $R^{\mathcal{B}}$ such that $(\mathcal{A}, R) \cong (\mathcal{B}, R^{\mathcal{B}})$. Once we know what element of B corresponds to $0 \in A$ under this isomorphism, we can use $R^{\mathcal{B}}$ to computably find the element of $\mathcal{B}$ that corresponds to $1 \in \mathcal{A}$, and then the one that corresponds to $2 \in A$, etc. Continuing this process, we get the desired computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$. □

The main question around computable categoricity is "what makes a structure computable categorical?" There has been a lot of work characterizing the computably categorical structures within certain classes of structures. See Table VIII.1.

Class	Condition for computable categoricity	Reference
Linear orderings	Finitely many pairs of adjacent elements	Dzgoev and Goncharov [**GD80**], Remmel [**Rem81**]
Boolean algebras	Finitely many atoms	Goncharov [**Gon75b**], La Roche [**LR78**]
$\mathbb{Q}$-vector spaces	Finite dimension	
Algebraically closed fields	Finite transcendence over prime subfield	Ershov [**Eršˇ77**]
Ordered abelian groups	Finite rank	Goncharov, Lempp, and Solomon [**GLS03**]
Trees of finite height	Finite type	Lempp, McCoy, R. Miller, and Solomon [**LMMS05**]
Torsion-free abelian groups	Finite rank	Nurtazin [**Nur74**]
Abelian p-groups	Either (i) $(\mathbb{Z}(p^\infty))^\ell \oplus G$ for $\ell \in \mathbb{N} \cup \{\infty\}$ and G finite, or (ii) $(\mathbb{Z}(p^\infty))^n \oplus (\mathbb{Z}_{p^k})^\infty \oplus G$ where G is finite, and $n, k \in \mathbb{N}$	Goncharov [**Gon80**], Smith [**Smi81**]

TABLE 1. The middle column describes a necessary and sufficient condition condition for a structure within the given class to be computably categorical. For the definitions of the relevant terms and the proofs, we refer the reader to the references given in the third column. Each case requires a different priority argument.

We do not expect such clean characterizations to be always possible. Downey, Kach, Lempp, Lewis-Pye, Montalbán, and Turetsky [**DKL**$^+$] showed that there is no structural characterization for the notion of computable categoricity. They did it by showing that the index set of the computably categorical structures is Π^1_1-complete. We do have structural characterizations if we consider variations of the notion of computable categoricity. For instance, recall that we have already introduced *uniformly computably categorical structures* in Section III.4 and proved they coincide with the effectively $\exists$-atomic ones. This chapter is dedicated to the non-uniform notions which are, in a sense, more natural. In particular, it is dedicated to the notion of relative computable categoricity and its connections to plain computable categoricity.

VIII.2. Relative computable categoricity

In this section, we give a purely structural characterization for the computational notion of relative computable categoricity.

DEFINITION VIII.2.1 ([**AKMS89**, Section 4][**Chi90**, Definition V.9]). Given $X \in 2^{\mathbb{N}}$, an X-computable structure $\mathcal{A}$ is *X-computably categorical* if there is an X-computable

isomorphism between any two X-computable copies of $\mathcal{A}$. A computable structure $\mathcal{A}$ is *relatively computably categorical* if it is X-computably categorical for all $X \in 2^{\mathbb{N}}$.

Equivalently, $\mathcal{A}$ is relatively computably categorical if, for every copy $\mathcal{B}$ (computable or not) of $\mathcal{A}$, there is an isomorphism between $\mathcal{B}$ and $\mathcal{A}$ that is computable in $D(\mathcal{B})$.

THEOREM VIII.2.2 (Ash, Knight, Manasse, Slaman [**AKMS89**, Theorem 4]; Chisholm [**Chi90**, Theorem V.10]). *Let $\mathcal{A}$ be a computable structure. The following are equivalent:*

(1) *$\mathcal{A}$ is relatively computably categorical.*
(2) *$(\mathcal{A}, \bar{a})$ is uniformly computably categorical for some $\bar{a} \in A^{<\mathbb{N}}$.*
(3) *$(\mathcal{A}, \bar{a})$ is effectively $\exists$-atomic for some $\bar{a} \in A^{<\mathbb{N}}$.*

PROOF. The equivalence between (2) and (3) was proved in Theorem III.4.2. To see that (2) implies (1), just notice that for any copy $\mathcal{B}$ of $\mathcal{A}$, one can non-uniformly pick the corresponding tuple $\bar{a}^{\mathcal{B}}$ so that $(\mathcal{B}, \bar{a}^{\mathcal{B}}) \cong (\mathcal{A}, \bar{a})$, and then use part (2) of Theorem III.4.2 to get a $D(\mathcal{B})$-computable isomorphism between them.

The interesting direction is the implication from (1) to (3), which shares some ideas with the proof of Theorem III.4.2. Assume $\mathcal{A}$ is relatively computably categorical. Out of this computational assumption, we need to build a syntactical object, namely a c.e. Scott family of $\exists$-definitions for the automorphism orbits of the tuples in $A^{<\mathbb{N}}$, over some parameters.

Let $g \colon \mathbb{N} \to \mathcal{A}$ be an enumeration of $\mathcal{A}$ that is *2-generic relative to the presentation of $\mathcal{A}$*: We have not defined this yet, so let us say that g is a 1-generic enumeration of the structure $(\mathcal{A}; \vec{K}^{\mathcal{A}}, D(\vec{\mathcal{A}}))$, where the diagram of $\mathcal{A}$ is being added as real in 2^ω (see Definition IV.2.1). Let $\mathcal{B}$ be the generic presentation obtained as the pull-back of $\mathcal{A}$ through g (as in Definition IV.2.3). Since $\mathcal{A}$ is relatively computably categorical, and $\mathcal{B} \cong \mathcal{A}$, there is a computable operator Γ such that $\Gamma^{D(\mathcal{B})}$ is an isomorphism from $\mathcal{B}$ to $\mathcal{A}$.

The first step is to get a tuple $\bar{q} \subseteq g$ which *forces* that $\Gamma^{D(\mathcal{B})}$ is an isomorphism in the following sense:

CLAIM VIII.2.2.1. There is a tuple $\bar{p} \subseteq g$ such that any tuple $\tilde{q} \supseteq \bar{p}$ can be extended to an enumeration $\tilde{g}$ with pull-back $\tilde{\mathcal{B}} = \tilde{g}^{-1}(\mathcal{A})$ so that $\Gamma^{D(\tilde{\mathcal{B}})}$ is an isomorphism from $\tilde{\mathcal{B}}$ to $\mathcal{A}$.

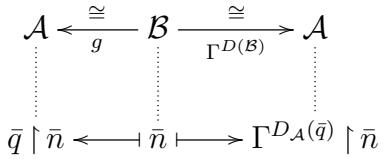
Let us leave the proof of the claim for later, and start by proving the theorem from it.

Given tuples $\bar{q} = (q_0, q_1, ...) \in A^{\leq \mathbb{N}}$ and $\bar{n} = (n_0, ..., n_\ell) \in \mathbb{N}^{<\mathbb{N}}$, we use $\bar{q} \restriction \bar{n}$ to denote $(q_{n_0}, ..., q_{n_\ell}) \in A^{|\bar{n}|}$. Since g and $\Gamma^{D(\mathcal{B})}$ are isomorphisms from $\mathcal{B}$ to $\mathcal{A}$, for every $\bar{n} \in \mathbb{N}^{<\mathbb{N}}$,

$$(\mathcal{A}, g \restriction \bar{n}) \;\cong\; (\mathcal{B}, \bar{n}) \;\cong\; (\mathcal{A}, \Gamma^{D(\mathcal{B})} \restriction \bar{n}).$$

Recall that if $\bar{q} \subseteq g$, then $D_{\mathcal{A}}(\bar{q}) \subseteq D(\mathcal{B})$ (Observation I.1.11). Therefore, if we have $\bar{q} \subseteq g$ so that $\Gamma^{D_{\mathcal{A}}(\bar{q})} \restriction \bar{n}$ converges (i.e., if $\Gamma^{D_{\mathcal{A}}(\bar{q})}(n_i) \downarrow$ for all $i \leq \ell$), then $\Gamma^{D_{\mathcal{A}}(\bar{q})} \restriction \bar{n}$ is automorphic to $\bar{q} \restriction \bar{n}$ as in the diagram below.

Here comes the key observation: the value of $\Gamma^{D_{\mathcal{A}}(\bar{q})} \restriction \bar{n}$ depends only on $D_{\mathcal{A}}(\bar{q}) \in 2^{<\mathbb{N}}$, while it determines the automorphism orbit of $\bar{q} \restriction \bar{n}$. Thus, informally: for $\bar{a} = \bar{q} \restriction \bar{n}$, the existential formula that says that $\bar{a}$ is part of a tuple $\bar{q}$ with this particular diagram defines the automorphism orbit of $\bar{a}$. Let us explain this in more detail. The key observation above can be formally stated as follows:



Claim VIII.2.2.2. If $\bar{q}, \tilde{q} \supseteq \bar{p}$ and $\Gamma^{D_{\mathcal{A}}(\bar{q})} \upharpoonright \bar{n} \downarrow$, then

$$D_{\mathcal{A}}(\bar{q}) = D_{\mathcal{A}}(\tilde{q}) \Rightarrow (\mathcal{A}, \bar{q} \upharpoonright \bar{n}) \cong (\mathcal{A}, \tilde{q} \upharpoonright \bar{n}).$$

To see this, from the previous claim we get an enumeration $\tilde{g} \supset \tilde{q}$ such that if $\tilde{\mathcal{B}} = \tilde{g}^{-1}(\mathcal{A})$, then $\Gamma^{D(\tilde{\mathcal{B}})}$ is an isomorphism. Then, using the observation from the diagram above, and that $\Gamma^{D_{\mathcal{A}}(\bar{q})} \upharpoonright \bar{n} = \Gamma^{D_{\mathcal{A}}(\tilde{q})} \upharpoonright \bar{n}$, we get that

$$(\mathcal{A}, \bar{q} \upharpoonright \bar{n}) \cong (\mathcal{B}, \bar{n}) \cong (\mathcal{A}, \Gamma^{D_{\mathcal{A}}(\bar{q})} \upharpoonright \bar{n}) = (\mathcal{A}, \Gamma^{D_{\mathcal{A}}(\tilde{q})} \upharpoonright \bar{n}) \cong (\tilde{\mathcal{B}}, \bar{n}) \cong (\mathcal{A}, \tilde{q} \upharpoonright \bar{n}),$$

as needed for the claim.

Fix a tuple $\bar{a}$; let us find a $\exists$-definition for the orbit of $\bar{a}$ under automorphisms of $\mathcal{A}$ that fix $\bar{p}$. Computably, search for a tuple $\bar{q}_{\bar{a}} \in A^{<\mathbb{N}}$ and a tuple $\bar{n}_{\bar{a}} \in \mathbb{N}^{<\mathbb{N}}$ such that

$$\bar{q}_{\bar{a}} \supseteq \bar{p}, \quad \bar{q}_{\bar{a}} \upharpoonright \bar{n}_{\bar{a}} = \bar{p}\bar{a} \quad \text{and} \quad \Gamma^{D_{\mathcal{A}}(\bar{q}_{\bar{a}})} \upharpoonright \bar{n}_{\bar{a}} \downarrow .$$

We will eventually find such tuples because one can always take $\bar{q}_{\bar{a}}$ to be a long enough initial segment of g and take $\bar{n}_{\bar{a}}$ so that $g \upharpoonright \bar{n}_{\bar{a}} = \bar{p}\bar{a}$. We claim that, for any tuple $\bar{b}$,

$$(\mathcal{A}, \bar{p}\bar{a}) \cong (\mathcal{A}, \bar{p}\bar{b}) \iff \exists \tilde{q} \left(\tilde{q} \supseteq \bar{p} \ \wedge \ \tilde{q} \upharpoonright \bar{n}_{\bar{a}} = \bar{p}\bar{b} \ \wedge \ D_{\mathcal{A}}(\tilde{q}) = D_{\mathcal{A}}(\bar{q}_{\bar{a}}) \right).$$

For the right-to-left direction, consider such a tuple $\tilde{q}$, and observe that $\bar{p}\bar{a}$ and $\bar{p}\bar{b}$ are automorphic by Claim VIII.2.2.2. For the left-to-right direction, let $\tilde{q}$ be the tuple that corresponds to $\bar{q}_{\bar{a}}$ through the automorphism mapping $\bar{p}\bar{a}$ to $\bar{p}\bar{b}$.

We can rewrite the right-hand side as an existential formula about $\mathcal{A}$ with parameters $\bar{p}$:

$$\varphi_{\bar{a}}(\bar{p}, \bar{x}) \ \equiv \ \exists \bar{y} \left(\bar{y} \supseteq \bar{p} \ \wedge \ \bar{y} \upharpoonright \bar{n}_{\bar{a}} = \bar{p}\bar{x} \ \wedge \ D(\bar{y}) = D_{\mathcal{A}}(\bar{q}_{\bar{a}}) \right),$$

(where $\bar{x}$ and $\bar{y}$ are replacing $\bar{b}$ and $\tilde{q}$, and where "$D(\bar{y}) = \sigma$" is shorthand for $\varphi^{\text{at}}_{\sigma}(\bar{y})$, as defined in I.1.8). The formula $\varphi_{\bar{a}}$ defines the orbit of $\bar{a}$ under automorphisms that fix $\bar{p}$. The set $\{\varphi_{\bar{a}} : \bar{a} \in A^{<\mathbb{N}}\}$ is thus the desired c.e. Scott family of $\exists$-formulas over $\bar{p}$.

What is now left now to prove is Claim VIII.2.2.1, that there is a $\bar{p}$ that forces $\Gamma^{D(\mathcal{B})}$ to be an isomorphism from $\mathcal{B}$ to $\mathcal{A}$.

Proof of Claim VIII.2.2.1. This is a standard forcing proof as we will see in Chapter **??**. For this particular forcing application, the techniques we have developed so far in Chapter IV are enough, as we did in Theorem V.3.6.

Recall that g is a 1-generic enumeration of $(\mathcal{A}; \vec{K}^{\mathcal{A}}, D(\mathcal{A}))$, and $\mathcal{B} = g^{-1}(\mathcal{A})$. Let us start by forcing $\Gamma^{D(\mathcal{B})}$ to behave correctly wherever it converges. For this, consider the set of strings which force it not to:

$$Q_1 = \{\bar{q} \in A^{\star} : \exists n < |\bar{q}| \ \left(\Gamma^{D_{\mathcal{A}}(\bar{q})} \upharpoonright n \downarrow \ \& \ D_{\mathcal{A}}(\Gamma^{D_{\mathcal{A}}(\bar{q})} \upharpoonright n) \neq D_{\mathcal{A}}(\bar{q} \upharpoonright n) \right)\}.$$

The set Q_1 is r.i. computable in $(\mathcal{A}; D(\mathcal{A}))$, and hence decided by some initial segment of the enumeration g.[1] No initial segment of g is in Q_1 because $\Gamma^{D_{\mathcal{A}}(\mathcal{B})}$ is an isomorphism, so there must be an initial segment $\bar{p}_1 \in A^\star$ of g such that no extension of $\bar{p}_1$ is in Q_1. This means that whenever $\bar{q} \in A^\star$ extends $\bar{p}_1$, if $\Gamma^{D_{\mathcal{A}}(\bar{p})}(n)\downarrow$, then $D_{\mathcal{A}}(\Gamma^{D_{\mathcal{A}}(\bar{q})} \upharpoonright n) = D_{\mathcal{A}}(\bar{q} \upharpoonright n)$.

Second, we force that $\Gamma^{D(\mathcal{B})}$ is total: For this, consider the set of strings which force $\Gamma^{D(\mathcal{B})}$ to be undefined at some $n \in \mathbb{N}$:

$$Q_2 = \{\bar{q} \in A^\star : \exists n \in \mathbb{N} \forall \bar{r} \in A^\star \left(\bar{r} \supseteq \bar{q} \to \Gamma^{D_{\mathcal{A}}(\bar{r})}(n)\uparrow\right)\}.$$

The set Q_2 is Σ^{c}_2 in $\mathcal{A}$, and hence r.i.c.e. in $(\mathcal{A}, \vec{K}^{\mathcal{A}})$ and decided by an initial segment of g.[2] We cannot have an initial segment of g in Q_2 because we would have that $\Gamma^{D(\mathcal{B})}(n)\uparrow$ for some n. So, for some initial segment $\bar{p}$ of g, we have that for every $\bar{q} \in A^\star$ extending $\bar{p}$ and every n, there is a $\bar{r} \in A^\star$ extending $\bar{q}$ for which $\Gamma^{D_{\mathcal{A}}(\bar{r})}(n)\downarrow$. We may assume $\bar{p} \supseteq \bar{p}_1$.

We claim that $\bar{p}$ is as wanted. Since $\bar{p}$ forces out of Q_2, for any $\tilde{q} \supseteq \bar{p}$, we can build a sequence $\tilde{q} \subseteq \bar{r}_1 \subseteq \bar{r}_2 \subseteq \bar{r}_3 \subseteq \cdots \in A^\star$ so that $\Gamma^{D_{\mathcal{A}}(\bar{r}_n)}(n)\downarrow$ for each n. If we also make sure that n is in the range of $\bar{r}_n$, we get an onto enumeration $\tilde{g} = \bigcup_n \bar{r}_n \colon \mathbb{N} \to A$, which satisfies that $\Gamma^{D(\tilde{\mathcal{B}})}$ is total, where $\tilde{\mathcal{B}} = \tilde{g}^{-1}(\mathcal{A})$. Since $\bar{p}$ forces out of Q_1, $\Gamma^{D(\tilde{\mathcal{B}})} \circ \tilde{g}^{-1} \colon \mathcal{A} \to \mathcal{A}$ must preserve diagrams and hence be an isomorphism. It follows that $\Gamma^{D(\tilde{\mathcal{B}})} \colon \tilde{\mathcal{B}} \to \mathcal{A}$ must be an isomorphism too. □

□

Exercise VIII.2.3. (Hard) (Originated after conversations between Harrison-Trainor, Hirschfeldt, Kalimullin, Melnikov, Montalbán, and Solomon.) The proof above uses the fact that $\mathcal{A}$ has a computable presentation. We can still have relatively computably categorical structures which don't have computable presentation: for any two copies $\mathcal{A}$ and $\mathcal{B}$, there is an isomorphism computable from $D(\mathcal{A}) \oplus D(\mathcal{B})$.

(a) Prove that Theorem VIII.2.2 is still true when $\mathcal{A}$ does not have computable copies. (In this case, the Scott family will have extra formulas that are not satisfied by any tuple in the structure.) Hint in footnote. [3]

(b) Show that in the setting of Theorem VIII.2.2, if the $\exists$-type of the parameters is c.e. in an oracle X, then $\mathcal{A}$ has a $\Pi^{\mathrm{c},X}_2$ Scott sentence.

(c) Show that $\mathcal{A}$ has enumeration degree given by the $\exists$-type of the parameters.

VIII.3. Categoricity on a cone

By the *Turing cone* above X, we mean the set $\{Y \in 2^{\mathbb{N}} : Y \geq_T X\}$. Sometimes, we will just call it a *cone*. A set $\mathcal{R} \subseteq 2^{\mathbb{N}}$ is said to be *degree invariant* if, for every $X, Y \in 2^{\mathbb{N}}$, if $X \in \mathcal{R}$ and $Y \equiv_T X$, then $Y \in \mathcal{R}$ too. Martin showed that every degree-invariant set of reals either contains a cone or is disjoint from a cone — if one assumes enough determinacy. This motivates viewing degree-invariant sets that contain

[1] Notice that $\Gamma^{D_{\mathcal{A}}(\bar{q})} \upharpoonright n$ is a tuple in $\mathbb{N}^{<\mathbb{N}}$, and we need to use $D(\mathcal{A})$ to figure out $D_{\mathcal{A}}(\Gamma^{D_{\mathcal{A}}(\bar{q})} \upharpoonright n) \in 2^{<\mathbb{N}}$ in this particular presentation of $\mathcal{A}$. When we wrote $D_{\mathcal{A}}(\Gamma^{D_{\mathcal{A}}(\bar{q})} \upharpoonright n) \neq D_{\mathcal{A}}(\bar{q} \upharpoonright n)$, it was a shorthand for $\neg\varphi^{\mathrm{at}}_\sigma(\bar{q} \upharpoonright n)$ for $\sigma = D_{\mathcal{A}}(\Gamma^{D_{\mathcal{A}}(\bar{q})} \upharpoonright n)$, where $\varphi^{\mathrm{at}}_\sigma$ is as in Observation I.1.8.

[2] To see that Q_2 is Σ^{c}_2, observe that $\{\langle \bar{q}, n\rangle : \forall \bar{r} \in A^\star \left(\bar{r} \supseteq \bar{q} \to \Gamma^{D_{\mathcal{A}}(\bar{r})}(n)\uparrow\right)\}$ is co-r.i.c.e. in $\mathcal{A}$ and hence Π^{c}_1-definable.

[3] You need to consider a generic presentation of $\mathcal{A} \sqcup \mathcal{A}$.

cones as *large*, and the ones disjoint from cones as *small*. It is not hard to show that countable intersections of large sets are still large, and countable unions of small sets are still small.

THEOREM VIII.3.1 (Martin [**Mar68**]). *If $\mathcal{R} \subseteq 2^{\mathbb{N}}$ is Borel and degree-invariant, then it either contains a cone or is disjoint from a cone.*

This is true for every degree-invariant set $\mathcal{R}$ if we assume the full axiom of determinacy. It is true for every degree-invariant analytic set $\mathcal{R}$ if we assume analytic determinacy.

We sketch this proof for the readers familiar with determinacy. The theorem is not relevant for the rest of the text, except as a motivation for Definition VIII.3.2. The reader not familiar with determinacy may freely skip it.

PROOF. Consider a game where Player I and Player II alternatively play binary bits $x_0, y_0, x_1, y_1, ... \in \{0, 1\}$ for infinitely many steps.

Player I	x_0		x_1		x_2	$\cdots$	$\cdots$	$\bar{x} \in 2^{\mathbb{N}}$
Player II		y_0		y_1		$\cdots$	$\cdots$	$\bar{y} \in 2^{\mathbb{N}}$

Player I wins the game if the sequence $\bar{x} \oplus \bar{y}$ belongs to $\mathcal{R}$, and Player II wins if it does not. By Borel determinacy, which Martin showed can be proved in ZFC [**Mar75**], one of the two players must have a winning strategy $s\colon 2^{<\mathbb{N}} \to 2$.

We claim that if Player I has a winning strategy, then the cone above s is included in $\mathcal{R}$; while if Player II has a winning strategy, the cone above s is disjoint from $\mathcal{R}$. Suppose s is a winning strategy for Player I, and let $\bar{y}$ be any real in the cone above s; we want to show that $\bar{y} \in \mathcal{R}$. Assume Player II plays $\bar{y}$, and let $\bar{x}$ be the response to $\bar{y}$ by a Player I following the strategy s. Since s is a winning strategy, we have that $\bar{x} \oplus \bar{y} \in \mathcal{R}$, and since $\bar{y} \geq_T s$, we have that $\bar{y} \geq_T \bar{x}$. Furthermore, $\bar{y} \equiv_T \bar{x} \oplus \bar{y}$. Since $\mathcal{R}$ is degree invariant, this implies that $\bar{y} \in \mathcal{R}$, as needed. The case where II has a winning strategy is analogous.

If $\mathcal{R}$ were analytic instead of Borel, we would need to use analytic determinacy to get this proof to work. Analytic determinacy does not follow from ZFC, but follows from weak large-cardinal hypotheses like the existence of sharps (Harrington [**Har78**]). If we do not want to impose any complexity assumption on $\mathcal{R}$, we would need the full axiom of determinacy. □

Suppose now we have a property of reals that is invariant under Turing equivalence. For instance, consider the set of $X \in 2^{\mathbb{N}}$ such that a given structure $\mathcal{A}$ is X-computably categorical. By Martin's theorem, this set must be either large or small — assuming analytic determinacy. In other words, either, relative to almost all oracles, $\mathcal{A}$ is computably categorical; or, relative to almost all oracles, $\mathcal{A}$ is not computably categorical.

DEFINITION VIII.3.2. A structure $\mathcal{A}$ is *computably categorical on a cone* if there is a $Y \in 2^{\mathbb{N}}$ such that $\mathcal{A}$ is X-computably categorical for all $X \geq_T Y$.

If $\mathcal{A}$ is a *natural* structure, a property like categoricity must easily relativize. Thus, for natural $\mathcal{A}$, the three notions of computable categoricity — plain, relative, and on a cone — must coincide. If we want to understand how computable categoricity works on "natural" structures, our best bet is to look at it on a cone. The reason is that on-a-cone

properties avoid counterexamples one can build by diagonalizing against all computable functions. This is because, now, one would have to diagonalize against all X-computable functions for almost all X, and there are continuum many of those. This is why it is often the case that on-a-cone properties have cleaner structural characterizations, as is the case for computable categoricity:

THEOREM VIII.3.3. *Let $\mathcal{A}$ be a countable structure. The following are equivalent:*
(1) *$\mathcal{A}$ is computably categorical on a cone.*
(2) *$\mathcal{A}$ is $\exists$-atomic over a finite set of parameters.*
(3) *$\mathcal{A}$ has a $\Sigma^{\mathtt{in}}_3$ Scott sentence.*

PROOF. The equivalence between the top two statements follows from the relativized version of Theorem VIII.2.2: Notice that $\mathcal{A}$ is computably categorical on a cone if and only if it is "relatively computably categorical" relative to some oracle X. The equivalence between the bottom two statements was proved in Lemma III.7.4. □

VIII.4. When relative and plain computable categoricity coincide

We saw in Table VIII.1 that computable categoricity can be completely understood within certain classes of structures, despite it being Π^1_1-complete in the general case. What is particular about the classes from Table VIII.1 is that, for them, plain and relative computable categoricity coincide. As we argued in Section VIII.3, for "natural" structures within any class, the two notions should also coincide. Goncharov proved that, under certain effectiviness conditions, computably categoricity is indeed well-behaved. His result is based on a theorem by Nurtazin that deals with yet another variation of the notion of computable categoricity.

DEFINITION VIII.4.1. Given an ω-presentation of a τ-structure $\mathcal{A}$, we define $ED(\mathcal{A}) \in 2^{\mathbb{N}}$, the *elementary diagram* of $\mathcal{A}$, the same way we defined its atomic diagram in I.1.2, but now considering all elementary first-order formulas instead of just the atomic ones. For $i \in \mathbb{N}$,

$$ED(\mathcal{A})(i) = \begin{cases} 1 & \text{if } \mathcal{A} \models \varphi^{\mathrm{el}}_i[x_j \mapsto j : j \in \mathbb{N}], \\ 0 & \text{otherwise,} \end{cases}$$

where $\{\varphi^{\mathrm{el}}_i : i \in \mathbb{N}\}$ is an effective listing of the elementary first-order τ-formulas.

An ω-presentation $\mathcal{A}$ is said to be *decidable* if $ED(\mathcal{A})$ is computable.

The notion of decidable structure is quite important in computable structure theory. If one were interested in studying theorems from model theory from a computational perspective, dealing with decidable structures may be more appropriate than with computable ones. The notions of computable categoricity and effective $\exists$-atomicity translate as follows:

DEFINITION VIII.4.2. $\mathcal{A}$ is *computably categorical for decidable copies* if there is a computable isomorphism between any two decidable copies of $\mathcal{A}$. $\mathcal{A}$ is *effectively atomic* if it has a c.e. Scott family of elementary first-order formulas (see Definition III.1.2).

Atomic structures are quite important in model theory, while $\exists$-atomic structure are relevant in computable structure theory. Exactly as in Theorem III.5.2, a structure is atomic if and only if every elementary type realized in the structure is supported by an

elementary formula, and the Scott family for the structure is given by the set of these supporting formulas. (In the case of full types, supported types are called *principal types*, and the supporting formulas are called *generating formulas*.)

THEOREM VIII.4.3 (Nurtazin [**Nur74**]). *Let $\mathcal{A}$ be a decidable structure. The following are equivalent:*

(1) *$\mathcal{A}$ is computably categorical for decidable copies.*
(2) *$\mathcal{A}$ is effectively atomic over a finite set of parameters.*

Let us highlight that, while in Theorem VIII.2.2 we could build a non-computable (generic) copy of $\mathcal{A}$ to apply relatively computable categoricity, we now need to build a decidable copy of $\mathcal{A}$ to apply the assumptions. Thus, generics will not be useful here, and the proof will thus have to be quite different.

PROOF. An easy back-and-forth argument shows that effective atomicity implies computable categoricity for decidable copies. We prove the other direction, which requires a finite-injury priority construction. The reader not familiar with priority construction should read Section VII.1 first.

The idea is to build a decidable copy $\mathcal{B}$ of $\mathcal{A}$ in a way that either there are no computable isomorphisms between $\mathcal{B}$ and $\mathcal{A}$, or there is a c.e. Scott family for $\mathcal{A}$. Thus, either part (1) fails or part (2) holds. There are two sets of requirements. First, for each e, we have:

Requirement $\mathbf{R_e}$: Either Φ_e is not an isomorphism from $\mathcal{B}$ to $\mathcal{A}$, or $\mathcal{A}$ has a c.e. Scott family over parameters.

If all these requirements are satisfied, then either one of them succeeds in building a Scott family and we get that $\mathcal{A}$ is effectively atomic over parameters, or all of them succeed in making sure no Φ_e is an isomorphism, and hence showing that $\mathcal{A}$ is not computably categorical for decidable copies.

As usual, we will build $\mathcal{B}$ by building a one-to-one enumeration $g\colon \mathbb{N} \to A$ and defining $\mathcal{B}$ as the pull-back $g^{-1}(\mathcal{A})$. The other set of requirements will guarantee that g is onto.

Requirement $\mathbf{P_e}$: The eth element of the ω-presentation $\mathcal{A}$ is in the range of g.

The requirements are listed in order of priority as usual: $P_0, R_0, P_1, R_1, ...$

We need to ensure that $\mathcal{B}$ is decidable despite g not being computable. To be able to speak in precise terms about this, we first need to define the elementary diagram of finite tuples the same way we did for atomic diagrams in Definition I.1.7. Given a tuple $\bar{a} = (a_0, ..., a_s) \in A^{<\mathbb{N}}$, we define the *elmentary diagram of $\bar{a}$ in $\mathcal{A}$, denoted $ED_{\mathcal{A}}(\bar{a})$,* as the string in $2^{|\bar{a}|}$ such that, for $i < |\bar{a}|$,

$$ED_{\mathcal{A}}(\bar{a})(i) = \begin{cases} 1 & \text{if } \mathcal{A} \models \varphi_i^{\text{el}}[x_j \mapsto a_j, j < s], \\ 0 & \text{otherwise.} \end{cases}$$

As in Observation I.1.11, we have that if g is an enumeration of $\mathcal{A}$, then

$$ED(g^{-1}(\mathcal{A})) = \bigcup_{k \in \mathbb{N}} ED_{\mathcal{A}}(g \upharpoonright k).$$

At each stage s of the construction, we will build an injective finite tuple $g_s \in A^{<\mathbb{N}}$. The g_s's will not form a nested sequence, so we will not be able to define g as their union. But the sequence will have a pointwise limit, and we will be able to define $g(i) = \lim_s g_s(i)$. We still need $\mathcal{B}$ to be decidable, though. So even if the g_s's are not nested, we require the strings $ED_{\mathcal{A}}(g_s) \in 2^{<\mathbb{N}}$ to be nested; that is, for all $s < t$, $ED_{\mathcal{A}}(g_s) \subseteq ED_{\mathcal{A}}(g_t)$. We will then have that

$$ED(\mathcal{B}) = \bigcup_{s \in \mathbb{N}} ED_{\mathcal{A}}(g_s) \in 2^{\mathbb{N}}$$

is computable.

Informally, the idea for satisfying R_e is as follows. R_e will try to define g_s so that, for some tuple $\bar{n} \in \mathbb{N}^{<\mathbb{N}}$, $\Phi_{e,s} \upharpoonright \bar{n}$ converges and disagrees with $g_s \upharpoonright \bar{n}$ on some elementary formula. This way, if R_e manages to preserve this tuple g_s so that it ends up being an initial segment of g, since g will be an isomorphism from $\mathcal{B}$ to $\mathcal{A}$, Φ_e will not. To do this, for every tuple $\bar{b} \in A^{<\mathbb{N}}$, once we see $\Phi_{e,s} \upharpoonright \bar{n} \downarrow = \bar{b}$ for some $\bar{n}$ and s, we enlist $\bar{b}$ as a possible candidate for diagonalization. From that point on, we will be looking for another tuple $\bar{c}$ disagreeing with $\bar{b}$ on some elementary formula, so we can try to define $g \upharpoonright \bar{n} = \bar{c}$. If we find it, R_e will require attention, and if attention is given to it at some stage t, it will define g_t so that $g_t \upharpoonright \bar{n} = \bar{c}$ and try to preserve this initial segment of g. If we do not find such a disagreeing tuple $\bar{c}$, the reason is that whatever commitment we made at stage s about $\bar{n}$ (namely that we must preserve $ED_{\mathcal{A}}(g_s)$) had to imply all other formulas about $\bar{b}$, and hence be a principal formula for the type of $\bar{b}$. If this happens for all tuples $\bar{b}$, we can build a Scott family for $\mathcal{A}$. To make sure this works, we will be monitoring that everything we later commit to regarding $\bar{b}$ (namely that we must preserve $ED_{\mathcal{A}}(g_t)$ for the new g_t) is implied by the potentially principal formula. If it is, then we are not really committing anything new; if it is not, we have found an opportunity to diagonalize.

What makes this more difficult is that R_e must respect the work done by other requirements. The same way R_e would like to preserve the initial segment of g he defined, higher-priority requirements will like to preserve their initial segments. At the beginning of stage $s + 1$, we will define $\bar{p}_e[s] \subseteq g_s$ to be the initial segment of g_s that has been defined by higher-priority requirements R_i for $i < e$ and P_i for $i \leq e$. R_e must preserve $\bar{p}_e[s]$; that is, it is only allowed to define g_{s+1} extending $\bar{p}_e[s]$. R_e must also preserve $ED(g_s)$; that is, it is only allowed to define g_{s+1} satisfying $ED(g_{s+1}) \supseteq ED(g_s)$.

The construction: At any given stage, the first few requirement will be *active* and the rest *inactive*. At each stage, the highest-priority inactive requirement will be *initialized* and become active. During the construction, requirements may be *canceled*, making them inactive again. At each stage, each active P_e requirement will have an *output* string $\bar{p}_e \in A^{<\mathbb{N}}$, and each active R_e requirement an *output* string $\bar{r}_e$. These strings will be nested, $\bar{p}_0 \subseteq \bar{r}_0 \subseteq \bar{p}_1 \subseteq \bar{r}_1 \subseteq \cdots$, and g_s will be the union of the output strings of the active requirements at stage s. These are not fixed strings, and the value of $\bar{p}_e$ or $\bar{r}_e$ may change throughout the stages. We write $\bar{p}_e[s]$ or $\bar{r}_e[s]$ if we want to highlight that we are referring to their values at stage s. We will show they will eventually reach a limit and stop changing.

Requirement $\mathbf{P_e}$ only acts the first time it is active after being inactive. If it is ever canceled, it will act again once it becomes active again. If it acts at stage $s + 1$,

its action consists of defining $g_{s+1} = g_s{}^\frown e$ (where e refers to the eth element of the ω-presentation $\mathcal{A}$). Well, that is if e is not in the range of g_s already, in which case we just define $g_{s+1} = g_s$. Once P_e acts, stage $s+1$ is over, and we move on directly to the next stage, $s+2$. We define the *output* of P_e to be $\bar{p}_e = g_{s+1}$, and this will stay this way unless P_e is later canceled. Since P_e will only act at a stage when no other requirement acts, we will have that $\bar{r}_{e-1}$, the output of R_{e-1}, is included in g_s. Thus, P_e indeed respects higher-priority requirements.

Requirement $\mathbf{R_e}$ works as follows. At each stage that is active, R_e may go through four *phases*:

- *waiting*,
- *internal calculations*,
- *requiring attention*, or
- *acting*.

We need to describe what R_e does in each of these phases. We leave the internal calculations phase for last.

Recall that $\bar{p}_e$ is the initial segment of g_s given by the output of the requirement of immediately higher priority, namely P_e. Once R_e has been activated, it will stay in the **waiting** phase until we reach a stage s at which $\Phi_{e,s} \upharpoonright |\bar{p}_e|$ converges. At the stages where $\Phi_{e,s} \upharpoonright |\bar{p}_e|$ does not converge, the requirement does not do anything, and we move on to consider the next active requirement. In the meantime, and until the requirement acts (if ever), its *output* is $\bar{r}_e = \bar{p}_e$. When we reach a stage s where $\Phi_{e,s} \upharpoonright |\bar{p}_e|$ converges, we let

$$\bar{a} = \Phi_{e,s} \upharpoonright |\bar{p}_e|$$

and move to the next phases of internal calculations and deciding if we require attention.

For tuples $\bar{n} \in \mathbb{N}^{<\mathbb{N}}$ and $\bar{p} \subseteq g_s$, we let $\psi_{\bar{n},g_s}(\bar{p},\bar{x})$ be the elementary formula describing the commitments we have made about $\bar{n}$ over $\bar{p}$ in $ED(g_s)$:

$$\psi_{\bar{n},g_s}(\bar{p},\bar{x}) \quad \equiv \quad \exists \bar{y}\ (\bar{y} \supseteq \bar{p}\ \wedge\ \bar{y} \upharpoonright \bar{n} = \bar{x}\ \wedge\ ED(\bar{y}) = \sigma), \qquad \text{where } \sigma = ED_{\mathcal{A}}(g_s) \in 2^{<\mathbb{N}},$$

and "$ED(\bar{y}) = \sigma$" is shorthand for what one would expect:

$$\text{“}ED(\bar{y}) = \sigma\text{”} \quad \equiv \quad \left(\bigwedge_{i:\sigma(i)=1} \varphi_i^{\mathrm{el}}(\bar{y}) \right) \wedge \left(\bigwedge_{i:\sigma(i)=0} \neg\varphi_i^{\mathrm{el}}(\bar{y}) \right).$$

Notice that $\mathcal{A} \models \psi_{\bar{n},g_s}(\bar{p}, g_s \upharpoonright \bar{n})$ with witness $\bar{y} = g_s$.

$\mathbf{R_e}$ requires attention if it finds an opportunity to diagonalize, that is, if it finds a tuple $\bar{n} \in \mathbb{N}^{<\mathbb{N}}$ of numbers greater than $|\bar{p}_e|$, a tuple $\bar{c} \in A^{<\mathbb{N}}$, and an elementary formula φ such that:

(1) $\Phi_{e,s} \upharpoonright \bar{n}$ converges,
(2) the tuples $\bar{p}_e{}^\frown \bar{c}$ and $\bar{a}{}^\frown \Phi_{e,s} \upharpoonright \bar{n}$ disagree on φ, and
(3) $\mathcal{A} \models \psi_{\bar{n},g_s}(\bar{p}_e, \bar{c})$.

After R_e requires attention, it may be allowed to act. Let $\bar{q}$ be the witness to $\mathcal{A} \models \psi_{\bar{n},g_s}(\bar{p}_e, \bar{c})$. That is,

$$\bar{q} \supseteq \bar{p}_e\ \wedge\ \bar{q} \upharpoonright \bar{n} = \bar{c}\ \wedge\ ED_{\mathcal{A}}(\bar{q}) = ED_{\mathcal{A}}(g_s).$$

The **action** of R_e is to define $g_{s+1} = \bar{q}$ and re-define $\bar{r}_e$, the *outcome* of R_e, also to be $\bar{q}$. If R_e is never canceled again, and g ends up being an isomorphism from $\mathcal{B}$ to $\mathcal{A}$ extending $\bar{r}_e$, R_e would have succeeded in diagonalizing against Φ_e, ensuring that Φ_e is not an isomorphism from $\mathcal{B}$ to $\mathcal{A}$. This is because, if Φ_e was an isomorphism, the tuples $\bar{p}_e{}^\frown\bar{c}$ and $\bar{a}{}^\frown\Phi_{e,s}\upharpoonright\bar{n}$ would have to be automorphic, contradicting they do not satisfy the same formulas. After this action, we cancel all the weaker-priority requirements, making them inactive, and finish stage $s+1$. R_e will not act again, and $\bar{r}_e$ will not change anymore, unless R_e is later canceled and re-initialized, in which case it will start over.

The **initial calculations of $\mathbf{R_e}$** are as follows. While R_e is looking for an instance to require attention, it will enumerate a set of formulas S, and hope it ends up being a Scott family for $\mathcal{A}$ over $\bar{p}_e$. Every time $\Phi_{e,s}$ converges on some new tuple $\bar{n}$ of numbers between $|\bar{p}_e|$ and $|g_s|$,

- define $\varphi_{\bar{n}}(\bar{x})$ be the formula $\psi_{\bar{n},g_s}(\bar{p}_e,\bar{x})$, and
- enumerate $\varphi_{\bar{n}}$ into S.

By doing this, R_e is betting that $\varphi_{\bar{n}}(\bar{x})$ is a formula generating the type of $\bar{b} = g\upharpoonright\bar{n}$ within $\mathcal{A}$ over $\bar{p}_e$. Later on, at each stage $u+1$ where a weaker-priority requirement R_i requires attention and wants to extend g_u to some tuple $\bar{h}$, we first check that

$$\mathcal{A} \models \forall\bar{x}(\varphi_{\bar{n}}(\bar{x}) \to \psi_{\bar{n},\bar{h}}(\bar{p}_e,\bar{x})).$$

If it does, we let the weaker-priority requirement do its thing and define $g_{u+1} = \bar{h}$. If it does not, R_e does not allow the weaker-priority requirement to act, as instead, R_e is in a position to require attention himself: We know there is a tuple $\bar{c}_1$ satisfying $\varphi_{\bar{n}}(\bar{c}_1) \wedge \psi_{\bar{n},\bar{h}}(\bar{p}_e,\bar{c}_1)$, namely $\bar{h}\upharpoonright\bar{n}$, and we know there is another tuple $\bar{c}_2$ that satisfies $\varphi_{\bar{n}}(\bar{c}_2) \wedge \neg\psi_{\bar{n},\bar{h}}(\bar{p}_e,\bar{c}_2)$ because the implication above does not hold. Let $\bar{c}$ be whichever of these two tuples disagrees with $\Phi_{e,s}\upharpoonright\bar{n}$ on $\psi_{\bar{n},\bar{h}}(\bar{p}_e,\bar{x})$. Since, at stage u, we checked that $\mathcal{A} \models \forall\bar{x}(\varphi_{\bar{n}}(\bar{x}) \to \psi_{\bar{n},g_u}(\bar{p}_e,\bar{x}))$, we have that $\mathcal{A} \models \psi_{\bar{n},g_u}(\bar{p}_e,\bar{c})$. R_e has now found the witnesses $\bar{n}$, $\bar{c}$, and $\psi_{\bar{n},\bar{h}}(\bar{p}_e,\bar{x})$ necessary to require attention at stage $u+1$.

Verifications: After a requirement is initialized, it will act at most once before it is re-initialized again, if ever. One can then prove, by induction on the list of requirements, that each requirement will eventually stop being canceled and will then eventually stop acting, and hence the next requirement will stop being canceled and then eventually stop acting, and so on. Since the outputs of the requirements only change when they act, we get that each $\bar{p}_e$ and $\bar{r}_e$ reaches a limit, and that g is the union of all these limits. Since each requirement P_e is eventually given the chance to act without being canceled again, we get that g is onto. Notice that g is one-to-one because each g_s is.

Let us now verify that each R_e is satisfied. Let s_e be the last stage in which P_e acted, so that R_e is never canceled after s_e. Suppose Φ_e is a computable isomorphism from $\mathcal{B}$ to $\mathcal{A}$. It must then be the case that R_e never requires attention after s_e, as otherwise, R_e would have acted and diagonalized against Φ_e, as we argued before. We claim that this implies that R_e is successful in making S into a Scott family. For each tuple $\bar{b} \in A^{<\mathbb{N}}$ disjoint from $\bar{p}_e$, there will be some $\bar{n}$ such that $g\upharpoonright\bar{n} = \bar{b}$, and there will be a first stage $s_{\bar{b}} > s_e$ at which $\Phi_{e,s_{\bar{b}}}\upharpoonright\bar{n}\downarrow$. At that stage, we enumerate $\varphi_{\bar{n}}(\bar{x})$ $(= \psi_{\bar{n},g_{s_{\bar{b}}}}(\bar{p}_e,\bar{x}))$ into S. We need to show that $\varphi_{\bar{n}}$ is indeed a generating formula for the elementary type of $\bar{b}$ over $\bar{p}_e$. First, notice that even if $g_s\upharpoonright\bar{n} \neq \bar{b}$, we still have that $\mathcal{A} \models \varphi_{\bar{n}}(\bar{b})$, because,

for every $t \geq s$, since $ED(g_t) \supseteq ED(g_s)$, we have that $\mathcal{A} \models \varphi_{\bar{n}}(g_t \upharpoonright \bar{n})$ as witnessed by $\bar{y} = g_t \upharpoonright |g_s|$. Since R_e never requires attention again, at every later stage $u > s_{\bar{b}}$, we have that

$$\mathcal{A} \models \forall \bar{x}(\varphi_{\bar{n}}(\bar{x}) \to \psi_{\bar{n},g_u}(\bar{p}_e, \bar{x})).$$

Every elementary formula $\varphi(\bar{p}_e, \bar{x})$ that is true of $\bar{b}$ in $\mathcal{A}$ will eventually be part of $ED_{\mathcal{A}}(g_u)$ for large enough u. Thus, φ is implied by $\psi_{\bar{n},g_u}(\bar{p}_e, \bar{x})$, and hence implied by $\varphi_{\bar{n}}(\bar{x})$. □

If we want to go back to the notion of computable categoricity (for computable copies), we can modify the proof above if we assume the two-quantifier theory of $\mathcal{A}$ is computable.

Definition VIII.4.4. A $\forall\exists$-*formula* is one of the form

$$\forall x_0 \forall x_1 ... \forall x_n \exists y_0 \exists y_1 ... \exists y_k \ \psi(\bar{x}, \bar{y}, \bar{z})$$

where ψ is finitary and quantifier-free. An ω-presentation $\mathcal{A}$ is $\forall\exists$-decidable if we can effectively decide all $\forall\exists$-formulas about the tuples of $\mathcal{A}$.

Theorem VIII.4.5 (Goncharov [**Gon75a**]). *If $\mathcal{A}$ is $\forall\exists$-decidable, then $\mathcal{A}$ is computably categorical if and only if it is effectively $\exists$-atomic over a finite set of parameters.*

Sketch of the proof. The proof is very similar to the proof above, but it requires being extra careful with the complexity of certain formulas at various steps of the construction. For this proof, we only need to preserve our usual atomic diagrams $D(g_s)$ instead of the elementary diagrams $ED(g_s)$. This will get us a computable ω-presentation $\mathcal{B}$. The formulas $\psi_{\bar{n},g_s}$ are now defined using $D(g_s)$ instead of $ED(g_s)$. Notice that $\psi_{\bar{n},g_s}$ is now an $\exists$-formula. When R_e is deciding if it requires attention, it now wants the tuples $\bar{p}_e{}^\frown\bar{c}$ and $\bar{a}{}^\frown\Phi_{e,s} \upharpoonright \bar{n}$ to disagree on some $\forall\exists$-formula, as that is what we can check computably. The key point where we used the decidability of $\mathcal{A}$ was during the initial-calculations phase to check whether

$$\mathcal{A} \models \forall \bar{x}(\varphi_{\bar{n}}(\bar{x}) \to \psi_{\bar{n},\bar{h}}(\bar{p}, \bar{x})).$$

This formula is now $\forall\exists$, which we can decide by the assumption on $\mathcal{A}$. However, we need to check a bit more. Let $\psi^{\forall}_{\bar{n},g_s}(\bar{p}_e, \bar{x})$ be the conjunction of all the $\forall$-formulas with indices less than $|g_s|$ that are true of $g_s(\bar{n})$ over $\bar{p}_e$. We also check that

$$\mathcal{A} \models \forall \bar{x}(\varphi_{\bar{n}}(\bar{x}) \to \psi^{\forall}_{\bar{n},\bar{h}}(\bar{p}, \bar{x})),$$

as this also gives us an opportunity to diagonalize. When we are verifying that R_e works, we only need to show that $\varphi_{\bar{n}}$ supports the $\forall$-type of $g \upharpoonright \bar{n}$ over $\bar{p}_e$. All these formulas are implied by $\psi^{\forall}_{\bar{n},g_u}(\bar{p}_e, \bar{x})$ for large enough u, so the proof is the same. □

Kudinov [**Kud96**] showed this result is sharp by building a $\forall$-decidable computably categorial structure that is not effectively $\exists$-atomic. Something we can say about $\forall$-decidable computably categorial structures is that they are effectively $\Sigma^{\mathfrak{c}}_2$-atomic, as proved by Downey, Kach, Lempp, and Turetksy [**DKLT13**, Theorem 1.13].

VIII.5. When relative and plain computable categoricity diverge

This section is dedicated to proving the following theorem.

THEOREM VIII.5.1 (Goncharov [**Gon77**, Theorem 4]). *There is a structure which is computably categorical, but not relatively so.*

This is an important theorem, and its proof illustrates a couple of techniques that are useful throughout the field. One is the use of families of sets to build structures with particular properties, which is a very common technique in the Russian school. The other one is the use of a finite-injury priority argument that is a bit more elaborated than the two we have seen before.

To prove Theorem VIII.5.1, we will build a c.e. family of sets $\mathcal{F} \subseteq \mathcal{P}(\mathbb{N})$, and then take the graph

$$\mathcal{G}^1_{\mathcal{F}} = \bigsqcup_{X \in \mathcal{F}} \mathcal{G}_X,$$

where $\mathcal{G}_X$ is the flower graph that consists of loops of size $n+3$, one for each $n \in X$, all with a common node. This is almost the same as the graph $\mathcal{G}^\infty_{\mathcal{F}}$ we considered in Observation V.4.2 and Lemma VI.1.10, with the difference that, in $\mathcal{G}^\infty_{\mathcal{F}}$, each $X \in \mathcal{F}$ is associated to infinitely many flower graphs $\mathcal{G}_X$ instead of just one as in $\mathcal{G}^1_{\mathcal{F}}$. Let us see how the relevant properties about structures translate to families.

DEFINITION VIII.5.2. A *computable Friedberg enumeration* of a family $\mathcal{F}$ is a c.e. set W whose columns are the sets in $\mathcal{F}$ without repetition.

Recall from Definition V.4.1 that a computable enumeration for a family $\mathcal{F}$ is a c.e. set W with $\mathcal{F} = \{W^{[i]} : i \in \mathbb{N}\}$, allowing for repeating columns. In a Friedberg enumeration, every set in $\mathcal{F}$ corresponds to exactly one column. In Observation V.4.2, we showed that $\mathcal{F}$ has a computable enumeration if and only if $\mathcal{G}^\infty_{\mathcal{F}}$ has a computable copy. As in Observation V.4.2, one can easily produce a computable Friedberg enumeration of $\mathcal{F}$ out of a computable ω-presentation of $\mathcal{G}^1_{\mathcal{F}}$, and vice versa.

DEFINITION VIII.5.3. A family $\mathcal{F} \subseteq \mathcal{P}(\mathbb{N})$ is *discrete* if there is a family S of finite sets such that, for each $A \in \mathcal{F}$, there is an $F \in S$ with $F \subseteq A$, and for each $F \in S$, there is a unique $A \in \mathcal{F}$ with $F \subseteq A$. We call such a set S a *separating family* for $\mathcal{F}$. We say that $\mathcal{F}$ is *effectively discrete* if $\mathcal{F}$ has a c.e. separating family.

LEMMA VIII.5.4. *Let $\mathcal{F} \subseteq \mathcal{P}(\mathbb{N})$ be a family with a c.e. enumeration. Then $\mathcal{G}^1_{\mathcal{F}}$ is effectively $\exists$-atomic if and only if $\mathcal{F}$ is effectively discrete.*

PROOF. Suppose $\mathcal{F}$ has a separating set S. We need to find $\exists$-formulas defining each node of $\mathcal{G}^1_{\mathcal{F}}$. Notice that each center of a flower graphs $\mathcal{G}_X$ is alone in its own automorphism orbit because each $\mathcal{G}_X$ appears only once in $\mathcal{G}^1_{\mathcal{F}}$. Also notice that if we have an $\exists$-formula defining the center of $\mathcal{G}_X$, we can find $\exists$-definitions for all the nodes in $\mathcal{G}_X$: We need to say that the node belongs to a loop of a certain size and that the loop also contains the center of $\mathcal{G}_X$. Thus, we will concentrate on enumerating $\exists$-definitions for the centers of the flower graphs. For each $X \in \mathcal{F}$, there is a finite set $A \in S$ such that X is the only set in $\mathcal{F}$ that contains A. Let $\varphi_X(x)$ be the formula that says that x is part of a loop of size $n+3$ for each $n \in A$. The center of G_X would be the only

element of $\mathcal{G}^1_{\mathcal{F}}$ satisfying that formula. Notice that if S is c.e., this produces a c.e. Scott family.

Suppose now that $\mathcal{G}^1_{\mathcal{F}}$ is $\exists$-atomic. For each X, let φ_X be the $\exists$-formula in the Scott family satisfied by the center of $\mathcal{G}_X$. Let A_X be a finite subset of X such that the center of a flower graph $\mathcal{G}_A$ also satisfies φ_X. Such an A_X must exist because if an $\exists$-formula is true of a relational structure, it is also true of a finite substructure. We claim that $\{A_X : X \in \mathcal{F}\}$ is a separating family for $\mathcal{F}$. We already argued that such an A_X exists for each X. If $A_X \subseteq Y$ for $Y \in \mathcal{F}$, then, since $\exists$-formulas are preserved under embeddings and $\mathcal{G}_{A_X}$ embeds in G_Y, we would have that φ_X holds of the center of G_Y too. Since φ_X defines the orbit of the center of G_X, we must have $X = Y$.

Notice that if we have a c.e. enumeration of $\mathcal{F}$, for each column X of the enumeration, we can effectively find φ_X within the given c.e. Scott family, and we then effectively find some A_X. □

Recall that a structure is relatively computable categorical if and only if it is effectively $\exists$-atomic over some parameters. So, we need to add the parameters to the previous lemma. We only need one direction.

Corollary VIII.5.5. *Let $\mathcal{F} \subseteq \mathcal{P}(\mathbb{N})$ be a discrete family with a c.e. enumeration. Then if $\mathcal{G}^1_{\mathcal{F}}$ is effectively $\exists$-atomic over parameters, $\mathcal{F}$ is effectively discrete.*

Proof. Let $\bar{p}$ be the parameters over which $\mathcal{G}^1_{\mathcal{F}}$ is effectively $\exists$-atomic. We can assume the elements of $\bar{p}$ are the centers of flowers, as from each $p \in \mathcal{G}^1_{\mathcal{F}}$ we can effectively find the center of the flower it belongs to, and vice-versa, we can effectively find p from the center of its flower. Since all flowers are completely independent, if we remove the flowers that contain $\bar{p}$ from $\mathcal{G}^1_{\mathcal{F}}$, we get a bouquet graph $\mathcal{G}^1_{\tilde{\mathcal{F}}}$ that is effectively $\exists$-atomic over no parameters. By the previous lemma, the corresponding family $\tilde{\mathcal{F}}$ is effectively discrete, and has a c.e. separating family $\tilde{S}$. Since $\mathcal{F}$ was discrete to begin with with, it has a separating family S, not necessarily c.e. Let S_0 be the finite subfamily of S that corresponds the flowers that contain $\bar{p}$, i.e., to the sets in $\mathcal{F} \setminus \tilde{\mathcal{F}}$. We then get that $\tilde{S} \cup S_0$ is a c.e. separating family for $\mathcal{F}$. □

Definition VIII.5.6. A *computable equivalence* between two computable enumerations, V and W, of a family $\mathcal{F}$ is a computable permutation f of $\mathbb{N}$ such that $W^{[n]} = V^{[f(n)]}$ for every n. When such a computable equivalence exists, we say that V and W are *computably equivalent*.

Lemma VIII.5.7. *$\mathcal{G}^1_{\mathcal{F}}$ is computably categorical if and only if $\mathcal{F}$ has only one Friedberg enumeration up to computable equivalence.*

Proof. We already know that computable ω-presentations of $\mathcal{G}^1_{\mathcal{F}}$ are in correspondence with c.e. Friedberg enumerations of $\mathcal{F}$. It is not hard to see that computable isomorphisms between ω-presentationsof $\mathcal{G}^1_{\mathcal{F}}$ are then in correspondence with computable equivalences between c.e. Friedberg enumerations of $\mathcal{F}$. □

Theorem VIII.5.1 now follows from the following lemma.

Lemma VIII.5.8 (Badaev [**Bad77**]). *There is a family $\mathcal{F} \subseteq \mathcal{P}(\mathbb{N})$ that is not effectively discrete and has only one computable Friedberg enumeration up to computable equivalence.*

PROOF. Let

$$E = \{0, 2, 4, 6, 8, ...\} \quad \text{and} \quad E_k = \{0, 2, 4, ..., 2k\} \cup \{2k + 1\}.$$

For each $n \in \mathbb{N}$, the family $\mathcal{F}$ will contain one set of the form $E \oplus \{n\}$, and either no or one set of the form $E_k \oplus \{n\}$. There will be no other sets in $\mathcal{F}$. We will build a computable Friedberg enumeration U of $\mathcal{F}$. To make $\mathcal{F}$ not effectively discrete, we have the following requirements:

Positive Requirement $\mathbf{P_e}$: W_e is not a separating family for $\mathcal{F}$.

To make sure $\mathcal{F}$ has a unique Friedberg enumeration, we have the following requirements:

Negative Requirement $\mathbf{N_e}$: If W_e is an Friedberg enumeration of $\mathcal{F}$, then W_e is computably equivalent to U.

The requirements are listed in decreasing order of priority as usual: $N_0, P_0, N_1, P_1, ...$ All the sets $E \oplus \{n\}$, for $n \in \mathbb{N}$, are enumerated into U from the beginning, say on the even columns of U. The sets $E_k \oplus \{n\}$ will be enumerated later on by the positive requirements P_e. Each P_e will act at most once, enumerating at most one such set. At each stage, each negative requirement N_i will impose a restraint on the P_e requirements of weaker priority by not allowing them to enumerate any set of the form $E_k \oplus \{n\}$ with $n < M_{i,s} \leq k$, where $M_{i,s}$ is a number defined by N_i at stage s of the construction. Each stage s of the construction starts with all the requirements N_i, for $i < s$, independently doing their own calculations and defining $M_{i,s}$. Then, all the requirements P_e for $e < s$ will independently do their thing as we describe below.

What makes these requirements "positive" and "negative," is that the P_e enumerate elements into U, while the N_e prevent elements from being enumerated.

The requirement P_e works as follows. Let $\{C_e : e \in \mathbb{N}\}$ be a computable partition of $\mathbb{N}$; for instance, let $C_e = \{\langle e, m\rangle : m \in \mathbb{N}\}$. The set C_e is reserved for requirement P_e. Suppose P_e has not been declared done yet. If we see a finite subset G with $\ulcorner G \urcorner \in W_e$ such that, for some $n \in C_e$ and some $m \in \mathbb{N}$, we have

- $G \subseteq \{0, 2, ..., 2m\} \oplus \{n\}$, and
- for each $i \leq e$, either $M_{i,s} \leq n$ or $m < M_{i,s}$,

then we add $E_m \oplus \{n\}$ to $\mathcal{F}$ (i.e., we enumerate it as a column in U), we declare P_e *done*, and we re-initialize all lower-priority N_i requirements. Recall that $M_{i,s}$ will be defined by N_i below. All we need to know for now about the sequence $M_{i,s}$ is that it is non-decreasing in s, and therefore that it converges to a limit — either to a number or to ∞. If W_e were indeed a separating family for F, then for every n, W_e would contain some set of the form $G = F \oplus \{n\}$ with $F \subseteq \{0, 2, ..., 2m\}$ for some m. Consider some $n \in C_e$ which is above $\lim_s M_{i,s}$ for all the $i \leq e$ for which the limit is finite. The corresponding m would eventually be below all the $M_{i,s}$ for all the $i \leq e$ for which the limit is infinite. P_e would then be allowed to act and enumerate $E_m \oplus \{n\}$ into $\mathcal{F}$. This contradicts that W_e is a separating family because G would be included in both $E \oplus \{n\}$ and $E_m \oplus \{n\}$ — P_e succeeds.

The requirement N_e works as follows. It will be *initialized* at stage $s + 1 = e$ and then will be re-initialized every time a higher-priority P_i requirement acts. Since each P_i acts at most once in the whole construction, there will be a point after which N_e will never be re-initialized again. Every time N_e is initialized, it starts building

a computable matching g_e between the columns of W_e to U by finite approximations $g_{e,0} \subseteq g_{e,1} \subseteq g_{e,2} \subseteq \cdots \to g_e$, with $g_{e,s} \in \mathbb{N}^{<\mathbb{N}}$. If it turns out that W_e is a Friedberg enumeration of $\mathcal{F}$ and that N_e is never re-initialized again, we have to make sure g_e is a computable equivalence between W_e and U. The rough idea is as follows: At each stage s, we will look at the columns of $W_{e,s}$ and $U[s]$, and hope there is an obvious way to match them. Whenever we see a set of the form $E_k \oplus \{n\}$ in both $W_{e,s}$ and in $U[s]$, we can safely match these columns through $g_{e,s}$. The problem arises when we need to match columns of the form $\{0, 2, ..., 2m\} \oplus \{n\}$: These columns may later grow in different ways and become $E_k \oplus \{n\}$ for some $k \geq m$ in W_e and $E \oplus \{n\}$ in U. To deal with this, N_e will impose a restraint not allowing sets of the form $E_k \oplus \{n\}$ for any $k \geq m$ to be enumerated into U by lower-priority requirements.

Let us start by defining an enumeration $\{V_{e,s} : s \in \mathbb{N}\}$ of $\mathcal{F}$ that is tidier than W_e. We do this by delaying the enumeration of certain elements, but in a way that if W_e is actually an enumeration of $\mathcal{F}$, then all elements of W_e eventually enter some $V_{e,s}$, so that $W_e = \bigcup_{s\in\mathbb{N}} V_{e,s}$. We want $V_{e,s}$ to satisfy the following properties for every $s \in \mathbb{N}$:

- $V_{e,s} \subseteq W_{e,s}$.
- Every non-empty column of $V_{e,s}$ is of the form $F \oplus \{n\}$ for some F and n.
- For every n, there are at most two such columns, one included in $E \oplus \{n\}$, and if there is a second one, it must be of the form $E_k \oplus \{n\}$.
- If $V_{e,s}$ contains a column of the form $E_k \oplus \{n\}$, then so does $U[s]$.

We can easily get such an enumeration $\{V_{e,s} : s \in \mathbb{N}\}$ just by enumerating the elements of a column of $W_{e,s}$ into $V_{e,s}$ only once the properties above are satisfied.

Let $M_{e,s}$ be the largest m such that, for every $n < m$, there is a column in $V_{e,s}$ containing $\{0, 2, 4, ..., 2m\} \oplus \{n\}$. Notice that $M_{e,s}$ is non-decreasing with s, and that if W_e is indeed an enumeration of $\mathcal{F}$, then $M_{e,s}$ converges to ∞. N_e imposes the following restraint on the lower-priority requirements:

> No set of the form $E_k \oplus \{n\}$ with $n < M_{e,s} \leq k$ can be enumerated into $\mathcal{F}$ at stage s.

At each stage s, we define a finite partial map $g_{e,s}$ matching columns in $V_{e,s}$ with columns in $U[s]$. We let $g_{e,s}(i) = j$ if and only if $V_{e,s}^{[i]}$ and $U^{[j]}[s]$ are of the forms $A \oplus \{n\}$ and $B \oplus \{n\}$ for the same n, respectively, and one the following holds:

(1) A and B are equal and of the form E_k for some k.
(2) $n < M_{e,s}$, $A \subseteq E$, $B = E$, and there is no column in $U[s]$ of the form $E_k \oplus \{n\}$ with $A \subseteq E_k$.

We claim that, unless N_e is re-initialized, $g_{e,s} \subseteq g_{e,s+1}$ for all s: If $g_{e,s}$ matches two columns of the form $E_k \oplus \{n\}$, those columns will still be matched in $g_{e,s+1}$. Suppose now $g_{e,s}$ matches two columns of the form $A \oplus \{n\}$ and $B \oplus \{n\}$ with $A, B \subseteq E$. We then must have that $n < M_{e,s}$, that $\{0, ..., 2M_{e,s}\} \subseteq A$, and there is no column in $U[s]$ of the form $E_k \oplus \{n\}$ with $A \subseteq E_k$. Because of the restraint imposed by N_e, no column of the form $E_k \oplus \{n\}$ with $k \geq M_{e,s}$ is enumerated into $U[s+1]$. (Notice that no higher-priority requirement acts, as we are assuming N_e is not re-initialized at s.) Thus, the column corresponding to A could not grow in $V_{e,s+1}$ to be of the form $E_k \oplus \{n\}$ by our assumption on $V_{e,s}$ that such columns must appear in $U[s+1]$ before they do in $V_{e,s+1}$ — this column is therefore still contained in $E \oplus \{n\}$. Since there is still no column in

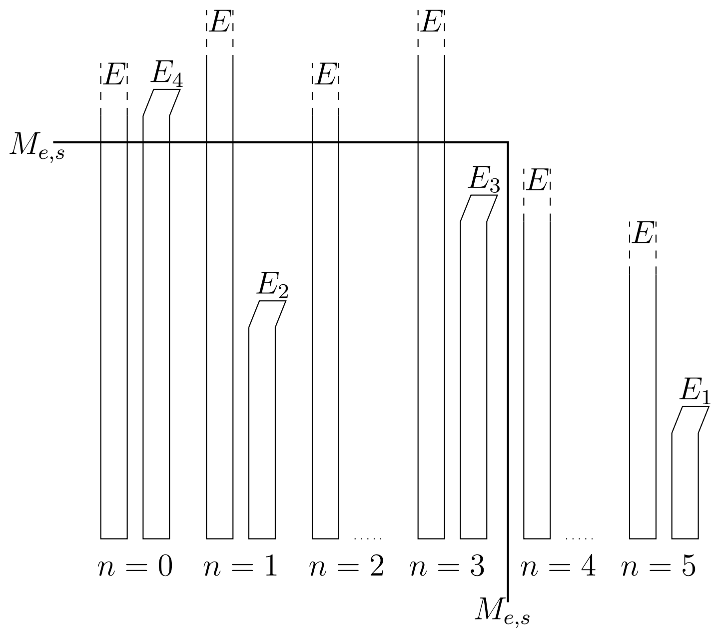


FIGURE 1. In this example, the restrain forbids us to enumerate a column of the form $E_k \oplus \{2\}$ for $k \geq M_e, s$ at stage s. That is, we cannot cross the horizontal $M_{e,s}$-line. So, for instance, the column that currently looks like $E \oplus \{2\}$ is not allowed to become of the form $E_k \oplus \{2\}$. The column $E_4 \oplus \{0\}$ crossing the line in the picture was enumerated before the current stage.

$U[s+1]$ of the form $E_k \oplus \{n\}$ with $A \subseteq E_k$, we get that the columns for A and B are matched again in the definition of $g_{e,s+1}$. This proves our claim, and we get that if N_e is never re-initialized again, $g_e = \bigcup_s g_{e,s}$ is a computable equivalence between W_e and U. $\square$

Notice that the family $\mathcal{F}$ is discrete, even if it is not effectively discrete. We thus get that $\mathcal{G}^1_{\mathcal{F}}$ is $\exists$-atomic and computably categorical on a cone. After Goncharov's result, there have been various other constructions of computably categorical structures which are not relatively so. For instance, Khoussainov, Semukhin, and Stephan [**KSS07**] built one without using a priority argument, using effective randomness instead. Their structure is not $\exists$-atomic over any finite set of parameters, so it is not computably categorical on a cone. Another example is due to Khoussainov and Shore [**KS98**, Theorem 4.2]. They built a computably categorical structure $\mathcal{A}$ such that, for each element $a \in \mathcal{A}$, the structure $(\mathcal{A}, a)$ is not computably categorical. The Khoussainov–Shore structure is not relatively computably categorical. This is because otherwise, it would remain relatively computably categorical if one added parameters.

EXERCISE VIII.5.9. (Open) What is the index set of computable categoricity on a cone?

EXERCISE VIII.5.10. Let $\mathcal{G}^1_{\mathcal{F}}$ be a bouquet graph as in Section VIII.5. Show that if the degree spectrum of $\mathcal{G}^1_{\mathcal{F}}$ has measure 1, then $\mathcal{G}^1_{\mathcal{F}}$ has a $0''$-computable copy.Hint in footnote.[4]

[4]Use Sacks' theorem that the measure of every non-trivial cone is 0.

CHAPTER IX

The Jump of a Structure

Given a structure $\mathcal{A}$, recall that we defined the complete r.i.c.e. relation $\vec{K}^{\mathcal{A}}$ by putting together all Σ^{c}_1-definable relations (Definition II.2.3):

$$\vec{K}^{\mathcal{A}} = \{\langle i, \bar{b}\rangle : \mathcal{A} \models \varphi^{\Sigma^{\mathsf{c}}_1}_{i,|\bar{b}|}(\bar{b})\} \subseteq \mathbb{N} \times A^{<\mathbb{N}},$$

where $\varphi^{\Sigma^{\mathsf{c}}_1}_{i,j}(\bar{x})$ is the ith Σ^{c}_1 τ-formula with j free variables. We then used this construction to define the jump of a relation $Q \subseteq A^{<\mathbb{N}}$ to be the relation $Q' = \vec{K}^{(\mathcal{A},Q)}$ (Definition II.2.7), and proved that this is an actual jump, that is, that $Q <^{\mathcal{A}}_{rT} Q'$ for all $Q \subseteq A^{<\mathbb{N}}$ (Corollary II.2.9). In this chapter, we consider this same construction, but view it as an operation from structures to structures.

Definition IX.0.11. Given an τ-structure $\mathcal{A}$, we define its jump to be the new structure obtained by adding the complete r.i.c.e. relation to it. That is, we let

$$\mathcal{A}' = (\mathcal{A}, \vec{K}^{\mathcal{A}}).$$

Thus, $\mathcal{A}'$ has the same domain as $\mathcal{A}$, but a larger vocabulary. It is a τ'-structure, where τ' consists of τ together with infinitely many new symbols naming the relations $K_{i,j} = \{\bar{b} \in A^j : \mathcal{A} \models \varphi^{\Sigma^{\mathsf{c}}_1}_{i,j}(\bar{b})\}$.

Notice that this definition is independent of the presentation of $\mathcal{A}$. The isomorphism type of $\mathcal{A}'$ depends only on the isomorphism type of $\mathcal{A}$. However it does depend — in an unessential way — on the Gödel numbering of the Σ^{c}_1 τ-formulas, in the same way as the Turing jump of a set depends on the Gödel numbering of the partial computable functions. Also notice that the extended language τ' is still a computable relational language.

Historical Remark IX.0.12. *The jump of structures has been introduced on various independent occasions over the last few years. Other definitions can be found in* [**Mor04, Bal06, Sos07, SS09, Puz09, Mon09, Stu09**]. *The definition we give here is from* [**Mon12**, Definition 5.1], *where the history of the different definitions is explained in more detail.*

Remark IX.0.13. Let us remark that the jump preserves effective bi-interpretabilty. That is, if $\mathcal{A}$ and $\mathcal{B}$ are effectively bi-interpretable, then so are $\mathcal{A}'$ and $\mathcal{B}'$. The interpretation maps are the same. All one has to observe is that the relation $\vec{K}^{\mathcal{A}}$ within the copy of $\mathcal{A}$ interpreted in $\mathcal{B}$ is r.i. computable in $\mathcal{B}'$ and vice versa. This is because Σ^{c}_2 formulas in $\mathcal{A}$ remain Σ^{c}_2 in the interpretation.

IX.1. The jump jumps — or does it?

The first thing to know about the jump is whether it is an actual jump, or whether there is a structure that is equivalent to its own jump. The answer is not straightforward

and depends on the notion of equivalence we use. For the strongest of the equivalences, namely effectively bi-interpretability, the jump does jump.

LEMMA IX.1.1. *No structure is Medvedev equivalent to its own jump. In particular, no structure is effectively bi-interpretable with its own jump.*

PROOF. We know from Lemma VI.1.9 that if $\mathcal{A}'$ were Medvedev reducible to $\mathcal{A}$, we would have $\exists\text{-}Th(\mathcal{A}') \leq_e \exists\text{-}Th(\mathcal{A})$. To show that it does not, we claim that $\exists\text{-}Th(\mathcal{A}')$ can enumerate the enumeration jump of $\exists\text{-}Th(\mathcal{A})$. The *enumeration jump* of a set X is defined to be

$$J(X) \oplus \overline{J(X)}, \quad \text{where} \quad J(X) = \{e : e \in \Theta_e^X\}$$

and $\{\Theta_e : e \in \mathbb{N}\}$ is an effective list of the enumeration operators as in page xii. A standard diagonalization argument shows that X cannot enumerate the set $\overline{J(X)}$. (If $\overline{J(X)}$ were enumeration reducible to X, we would have $\overline{J(X)} = \Theta_e^X$ for some e. We would then have that $e \in \overline{J(X)} \iff e \in \Theta_e^X \iff e \in J(X)$.)

Let us now prove our claim that $\overline{J}(\exists\text{-}Th(\mathcal{A})) \leq_e \exists\text{-}Th(\mathcal{A}')$. For $e \in \mathbb{N}$, $e \in \overline{J}(\exists\text{-}Th(\mathcal{A}))$ if and only if, for every finite set $D \subset \mathbb{N}$ with $\langle \ulcorner D \urcorner, e\rangle \in \Theta_e$, $D \not\subseteq \exists\text{-}Th(\mathcal{A})$. That is,

$$e \in \overline{J}(\exists\text{-}Th(\mathcal{A})) \iff \mathcal{A} \models \bigwedge\!\!\!\bigwedge_{\substack{D\subseteq\mathbb{N}\\ \langle D,e\rangle\in\Theta_e}} \neg \bigwedge_{i\in D} \varphi_i^{\exists},$$

where $\varphi_i^{\exists}$ is the ith existential τ-sentence. The right-hand side is a Π_1^{c} sentence about $\mathcal{A}$, and hence decided in the quantifier-free theory of $\mathcal{A}'$. So we even get that $\overline{J}(\exists\text{-}Th(\mathcal{A})) \leq_m \exists\text{-}Th(\mathcal{A}')$. □

The question of whether there is a structure that is Muchnik equivalent to its own jump turned out to be quite interesting.

THEOREM IX.1.2 (Puzarenko [**Puz11**], Montalbán [**Mon13c**]). *There is a structure $\mathcal{A}$ for which $\mathcal{A}'$ is effectively interpretable in $\mathcal{A}$ using one element of A as a parameter. In particular, this structure $\mathcal{A}$ is Muchnik equivalent to its own jump.*

The techniques used for this proof go beyond the scope of this book; we refer the reader to the original papers by Puzarenko [**Puz11**] and Montalbán [**Mon13c**]. These two proofs are quite different. Montalbán uses the existence of $0^\sharp$, and is a paragraph long once the definition of $0^\sharp$ is understood. Puzarenko's proof works inside ZFC, uses admissibility theory, and is much more complicated. Both proofs build an ill-founded ω-model $\mathcal{A}$ of $ZF^- + V = L$ where, for some ordinal α of the model, $(L_\alpha)^{\mathcal{A}} \cong \mathcal{A}$.

More surprising than the theorem itself is the complexity necessary to prove it. The theorem below shows that building a structure that is Muchnik equivalent to its own jump requires infinitely many uses of the power-set axiom. Puzarenko's proof of Theorem IX.1.2 uses KP plus $\omega_1^{CK} + 1$ iterations of the power-set axiom. There is still a gap as to how many iterates of the power-set axiom are needed to prove Theorem IX.1.2.

THEOREM IX.1.3 (Montalbán [**Mon13c**]). *Higher-order arithmetic cannot prove that there exists a structure $\mathcal{A}$ that is Muchnik equivalent to its own jump. Higher-order arithmetic refers to the union of nth-order arithmetic for all $n \in \mathbb{N}$.*

One of the main steps to prove this theorem is to show that, for such a structure $\mathcal{A}$, the set of reals coded by $\mathcal{A}$, namely $\{X \subseteq \mathbb{N} : \vec{X} \text{ is rice in } \mathcal{A}\}$, is the second-order part of an ω-model of full second-order arithmetic. Generalizing this to higher orders, Montalbán proved that the ω-jump of any presentation of $\mathcal{A}$ computes a countably coded ω-model of higher-order arithmetic.

Exercise IX.1.4. (Hard) Show that if $\mathcal{A}$ and $\mathcal{A}'$ are Muchnik equivalent, then $(\omega; \mathcal{M})$ where $\mathcal{M} = \{X \subseteq \mathbb{N} : \vec{X} \text{ is rice in } \mathcal{A}\}$, is a model of second order arithmetic. Hint in footnote.[1]

IX.2. The jump-inversion theorems

Friedberg's jump-inversion theorem [**Fri57a**] says that every Turing degree above $0'$ is the jump of some other degree. There are a couple of different ways in which one could generalize Friedberg theorem to the jump of structures. We call them the first and second jump-inversion theorems.

IX.2.1. The first jump-inversion theorem. This theorem is a generalization of the Friedberg jump-inversion theorem to the semi-lattice of structures ordered by effective interpretability.

Theorem IX.2.1 (Soskova, Stukachev). *For every structure $\mathcal{A}$ which codes $\overline{0'}$ (i.e., $\overrightarrow{0'}$ is r.i. computable in $\mathcal{A}$), there is a structure $\mathcal{C}$ whose jump is effectively bi-interpretable with $\mathcal{A}$.*

Proof. We proved in Theorem VI.2.9 that every structure is effectively bi-interpretable with a graph. Therefore, we may assume $\mathcal{A}$ is a graph $(A; E)$ with domain A and edge relation E. The key idea behind this proof is the following: If we are given a linear ordering isomorphic to either ω or ω^*, deciding which one is the case is a Δ^0_2 complete question. We will thus define $\mathcal{C}$ by removing the edge relation E and instead attaching to each pair of elements of A one of these two linear orderings, depending on whether there is an edge between the two elements or not.

We define $\mathcal{C}$ as $(C; A, R)$, where A is a unary relation and R a 4-ary relation. The domain C of $\mathcal{C}$ consists of the disjoint union of the domain A of $\mathcal{A}$ and another set B, and we use the unary relation A to identify the elements of A. We define the 4-ary relation

$$R \subseteq A \times A \times B \times B$$

so that is satisfies the following: If we let $B_{a,b} = \{c \in B : R(a, b, c, c)\}$, and $R_{a,b} = \{(c, d) \in B^2 : R(a, b, c, d)\}$, then $(B_{a,b}; R_{a,b})$ is a linear ordering isomorphic to either ω or ω^*, and it is isomorphic to ω if and only if $(a, b) \in E$.

$\mathcal{C}$ can be easily effectively interpreted in $\mathcal{A}$ as follows. Let $B = A^2 \times \mathbb{N}$ and let $C = A \cup B$ (coded as a subset of $A^{<\mathbb{N}}$ as in Definition II.1.23). Then define R as follows:

$$\begin{aligned} R = \{(a, b, (a, b, n), (a, b, m)) \in A^2 \times B^2 : \text{ for } (a, b) \in E \ \ \& \ \ n \leq m\} \\ \cup \{(a, b, (a, b, n), (a, b, m)) \in A^2 \times B^2 : \text{ for } (a, b) \in A^2 \smallsetminus E \ \ \& \ \ n \geq m\}. \end{aligned}$$

[1] You need to show the comprehension axioms hold in $(\omega; \mathcal{M})$. First note that every set in $\mathcal{M}$ can be named by an index for a c.e. operator and a tuple $\bar{p}$ using Corollary II.1.21. Then, translate set-quantification in the model to quantification over numbers and tuples from $\mathcal{A}$.

To show that this is actually an effective interpretation of $\mathcal{C}'$, and not just of $\mathcal{C}$, we need to show that $\vec{K}^{\mathcal{C}}$ (viewed as a relation in $A^{<\mathbb{N}}$) is r.i. computable within $\mathcal{A}$. To see this, fix an ω-presentation of $\mathcal{A}$. The construction above then gives us an ω-presentation of $\mathcal{C}$. Use Friedberg's jump-inversion theorem to get an oracle $X \in 2^{\mathbb{N}}$ such that $X' \equiv_T D(\mathcal{A})$ (that is where we use that $\mathcal{A}$ codes $\overline{0'}$). We will now construct a second copy, $\tilde{\mathcal{C}}$, of $\mathcal{C}$ that is computable in X. For each $(a,b) \in A^2$, X' knows if $(a,b) \in E$ or not, and hence computably in X, we can uniformly build a linear ordering $\tilde{\mathcal{B}}_{a,b}$ such that

$$\tilde{\mathcal{B}}_{a,b} \cong \begin{cases} (\mathbb{N}; \leq) & \text{if } (a,b) \in E, \\ (\mathbb{N}; \geq) & \text{if } (a,b) \notin E. \end{cases}$$

To do this, if $f(a,b,s)$ is an X-computable function such that $\lim_{s\in\mathbb{N}} f(a,b,s) = 1$ if $(a,b) \in E$ and $\lim_{s\in\mathbb{N}} f(a,b,s) = 0$, then we can define $\tilde{\mathcal{B}}_{a,b} = (\mathbb{N}; \leq_{\tilde{\mathcal{B}}_{a,b}})$ by

$$s \leq_{\tilde{\mathcal{B}}_{a,b}} r \iff \big(s \leq_{\mathbb{N}} r \ \ \& \ \ f(a,b,r) = 1\big) \vee \big(r \leq_{\mathbb{N}} s \ \ \& \ \ f(a,b,s) = 0\big).$$

We let the reader verify this ordering is as needed. We then define $\tilde{\mathcal{C}}$ by putting together $\mathcal{A}$ and disjoint copies of all the $\tilde{\mathcal{B}}_{a,b}$ for $(a,b) \in A^2$ and defining $\tilde{R}(a,b,n,m) \iff n,m \in \tilde{\mathcal{B}}_{a,b}$ & $n \leq_{\mathcal{B}_{a,b}} m$. An important point is that $D(\mathcal{A})$ can compute an isomorphism between $\mathcal{C}$ and $\mathcal{C}'$. This is because X' can compute isomorphisms between $\tilde{\mathcal{B}}_{a,b}$ and $\mathcal{B}_{a,b}$ for all $(a,b) \in A^2$. Since $D(\tilde{\mathcal{C}}) \leq_T X$, we have that $\vec{K}^{\tilde{\mathcal{C}}}$ is computable in X', and hence in $D(\mathcal{A})$. Going through the isomorphism between $\tilde{\mathcal{C}}$ and $\mathcal{C}$, we get that $\vec{K}^{\mathcal{C}}$ is also computable in $D(\mathcal{A})$. Since this worked for every ω-presentation of $\mathcal{A}$, we have that $\vec{K}^{\mathcal{C}}$ is r.i. computable in $\mathcal{A}$. This proves that we have an effective interpretation of $\mathcal{C}'$ in $\mathcal{A}$.

The effective interpretation of $\mathcal{A}$ within $\mathcal{C}'$ is even more direct. The domain of the interpretation is, of course, A itself, as identified by the relation A within $\mathcal{C}$. Notice that E is now r.i. Δ^0_2 in $\mathcal{C}$. This is because, to decide if $(a,b) \in A^2$, we need to decide whether $\mathcal{B}_{a,b} \cong \omega$ or $\mathcal{B}_{a,b} \cong \omega^*$. For this, we need to decide whether there exists an element in $\mathcal{B}_{a,b}$ without predecessors, or there exists an element without successors — both are Σ^0_2 questions.

The last step is to check that these two effective interpretations form an effective bi-interpretation; i.e., that the composition of the isomorphisms are r.i. computable in their respective structures. First, notice that the interpretation of $\mathcal{A}$ inside $\mathcal{C}$ inside $\mathcal{A}$ is the identity, and hence obviously r.i. computable in $\mathcal{A}$. Second, for the interpretation of $\mathcal{C}$ inside $\mathcal{A}$ inside $\mathcal{C}$, the A-part stays the same. The copies of $\mathcal{B}_{a,b}$ are not the same, but since they are isomorphic to either ω or ω^*, the isomorphism between them can be computed within a jump of $\mathcal{C}$. □

Historical Remark IX.2.2. *For the case of Muchnik equivalence, this theorem was proved independently on two occasions. One is due to Goncharov, Harizanov, Knight, McCoy, R. Miller and Solomon by essentially the same proof we gave above* [**GHK+05**, Lemma 5.5 for $\alpha = 2$], *although they were not considering jumps of structures. The other is due to Alexandra Soskova* [**Sos07, SS09**]. *Her construction is quite different and uses Marker extensions. Stukachev* [**Stu10, Stu**] *proved that Soskova's constructions actually gives effective interpretations instead of just Muchnik reductions.*

IX.2.2. The second jump-inversion theorem. This second jump-inversion theorem is not a generalization of the usual jump-inversion theorem to a more general class

of degrees, but a generalization in the sense that, given $X \in 2^{\mathbb{N}}$, it gives $Y \in 2^{\mathbb{N}}$ with $Y' \equiv_T X$ and some extra properties.

THEOREM IX.2.3 (Soskov). *If X computes a copy of $\mathcal{A}'$, then there is a set Y that computes a copy of $\mathcal{A}$ satisfying $Y' \equiv_T X$.*

PROOF. By Lemma IV.2.2, there is a 1-generic enumeration g of $\mathcal{A}$ computable in $\vec{K}^{\mathcal{A}}$, and hence in X. Let $\mathcal{B} = g^{-1}(\mathcal{A})$ and $Z = D(\mathcal{B})$. Since $\vec{K}^{\mathcal{B}} = g^{-1}(\vec{K}^{\mathcal{A}})$, we have that

$$\vec{K}^{\mathcal{B}} \leq_T \vec{K}^{\mathcal{A}} \leq_T X.$$

Since $\mathcal{B}$ is 1-generic,

$$\vec{K}^{\mathcal{B}} \equiv_T D(\mathcal{B})' = Z',$$

as proved in Lemma IV.3.4. Thus, $Z' \leq_T X$. By the relativized Friedberg's theorem, there is a $Y \in 2^{\mathbb{N}}$ such that $Z \leq_T Y$ and $Y' \equiv_T X$. This Y computes $\mathcal{B}$, a copy of $\mathcal{A}$. □

As a corollary, we get that the degree spectrum of the jump of a structure is what it should be: the set of jumps of the degrees in the spectrum of the original structure.

COROLLARY IX.2.4. *For every structure $\mathcal{A}$,*

$$DgSp(\mathcal{A}') = \{X \in 2^{\mathbb{N}} : X \geq_T Y' \text{ for some } Y \in DgSp(\mathcal{A})\}.$$

PROOF. For the right-to-left inclusion, it is clear that if $X \geq_T Y'$ for some $Y \in DgSp(\mathcal{A})$, then X computes a copy of $\mathcal{A}'$. For the left-to-right inclusion, if X computes a copy of $\mathcal{A}'$, then by the theorem, there is a $Y \in DgSp(\mathcal{A})$ such that $X \geq_T Y'$. □

HISTORICAL REMARK IX.2.5. *Theorem IX.2.3 was first introduced by Soskov at a talk at the LC'02 in Munster; a full proof then appeared in* [**SS09**]. *It was also independently proved in* [**Mon09**].

Part 2

Transfinite Topics

CHAPTER X

Infinitary Logic

CHAPTER XI

Hyperarithmetic theory

CHAPTER XII

Computable Infinitary Logic

CHAPTER XIII

Forcing

XIII.1. Forcing an infinitary formula

XIII.2. Forcing equals truth and the definability of forcing

XIII.3. The Ash-Knight-Manasse-Slaman–Chisholm theorem

XIII.4. Relative Δ^0_α-categoricity

CHAPTER XIV

α-priority aguments

Part 3

Classes of Structures

CHAPTER XV

Axiomatizable classes in descriptive set theory

CHAPTER XVI

Comparing classes of structures

CHAPTER XVII

Σ-small classes

Index

Bibliography

[ACK+] U. Andrews, M. Cai, I. Kalimullin, S. Lempp, J. Miller, and A. Montalbán. The complements of lower cones of degrees and the degree spectra of structures. To appear in the Journal of Symbolic Logic.

[AK00] C.J. Ash and J. Knight. *Computable Structures and the Hyperarithmetical Hierarchy*. Elsevier Science, 2000.

[AKMS89] Chris Ash, Julia Knight, Mark Manasse, and Theodore Slaman. Generic copies of countable structures. *Ann. Pure Appl. Logic*, 42(3):195–205, 1989.

[AM15] Uri Andrews and Joseph S. Miller. Spectra of theories and structures. *Proc. Amer. Math. Soc.*, 143(3):1283–1298, 2015.

[Bad77] S. A. Badaev. Computable enumerations of families of general recursive functions. *Algebra i Logika*, 16(2):129–148, 249, 1977.

[Bal06] V. Baleva. The jump operation for structure degrees. *Arch. Math. Logic*, 45(3):249–265, 2006.

[Bar75] Jon Barwise. *Admissible sets and structures*. Springer-Verlag, Berlin, 1975. An approach to definability theory, Perspectives in Mathematical Logic.

[BT79] V. Ja. Beljaev and M. A. Taĭclin. Elementary properties of existentially closed systems. *Uspekhi Mat. Nauk*, 34(2(206)):39–94, 1979.

[Chi90] John Chisholm. Effective model theory vs. recursive model theory. *J. Symbolic Logic*, 55(3):1168–1191, 1990.

[CHS07] Wesley Calvert, Valentina Harizanov, and Alexandra Shlapentokh. Turing degrees of isomorphism types of algebraic objects. *J. Lond. Math. Soc. (2)*, 75(2):273–286, 2007.

[Coo04] S. Barry Cooper. *Computability theory*. Chapman & Hall/CRC, Boca Raton, FL, 2004.

[Cut80] Nigel Cutland. *Computability*. Cambridge University Press, Cambridge-New York, 1980. An introduction to recursive function theory.

[DHK+07] Downey, Hirschfeldt, Kach, Lempp, A. Montalbán, and Mileti. Subspaces of computable vector spaces. *Journal of Algebra*, 314(2):888–894, August 2007.

[DK92] Rodney Downey and Julia F. Knight. Orderings with αth jump degree $\mathbf{0}^{(\alpha)}$. *Proc. Amer. Math. Soc.*, 114(2):545–552, 1992.

[DKL+] R. Downey, A. Kach, S. Lempp, A.E.M. Lewis-Pye, A. Montalbán, and D. Turetsky. The complexity of computable categoricity. Submitted for publication.

[DKLT13] Rodney G. Downey, Asher M. Kach, Steffen Lempp, and Daniel D. Turetsky. Computable categoricity versus relative computable categoricity. *Fund. Math.*, 221(2):129–159, 2013.

[End11] Herbert B. Enderton. *Computability theory*. Elsevier/Academic Press, Amsterdam, 2011. An introduction to recursion theory.

[Erš77] Ju. L. Eršov. Theorie der Numerierungen. III. *Z. Math. Logik Grundlagen Math.*, 23(4):289–371, 1977. Translated from the Russian and edited by G. Asser and H.-D. Hecker.

[Ers96] Yuri L. Ershov. *Definability and computability*. Siberian School of Algebra and Logic. Consultants Bureau, New York, 1996.

[FF09] Ekaterina B. Fokina and Sy-David Friedman. Equivalence relations on classes of computable structures. In *Mathematical theory and computational practice*, volume 5635 of *Lecture Notes in Comput. Sci.*, pages 198–207. Springer, Berlin, 2009.

[FFH+12] E. B. Fokina, S. Friedman, V. Harizanov, J. F. Knight, C. McCoy, and A. Montalbán. Isomorphism and bi-embeddability relations on computable structures. *Journal of Symbolic Logic*, 77(1):122–132, 2012.

[FK] Marat Faizrahmanov and Iskander Kalimullin. Limitwise monotonic sets of reals. submitted for publication.

[FKM09] Andrey Frolov, Iskander Kalimullin, and Russell Miller. Spectra of algebraic fields and subfields. In *Mathematical theory and computational practice*, volume 5635 of *Lecture Notes in Comput. Sci.*, pages 232–241. Springer, Berlin, 2009.

[Fri57a] Richard Friedberg. A criterion for completeness of degrees of unsolvability. *J. Symb. Logic*, 22:159–160, 1957.

[Fri57b] Richard M. Friedberg. Two recursively enumerable sets of incomparable degrees of unsolvability (solution of Post's problem, 1944). *Proc. Nat. Acad. Sci. U.S.A.*, 43:236–238, 1957.

[GD80] S. S. Gončarov and V. D. Dzgoev. Autostability of models. *Algebra i Logika*, 19(1):45–58, 132, 1980.

[GHK+05] Sergey Goncharov, Valentina Harizanov, Julia Knight, Charles McCoy, Russell Miller, and Reed Solomon. Enumerations in computable structure theory. *Ann. Pure Appl. Logic*, 136(3):219–246, 2005.

[GLS03] Sergey S. Goncharov, Steffen Lempp, and Reed Solomon. The computable dimension of ordered abelian groups. *Adv. Math.*, 175(1):102–143, 2003.

[Gon75a] S. S. Gončarov. Selfstability, and computable families of constructivizations. *Algebra i Logika*, 14(6):647–680, 727, 1975.

[Gon75b] S. S. Goncharov. Some properties of the constructivization of boolean algebras. *Sibirskii Matematicheskii Zhurnal*, 16(2):264–278, 1975.

[Gon77] S. S. Gončarov. The number of nonautoequivalent constructivizations. *Algebra i Logika*, 16(3):257–282, 377, 1977.

[Gon80] Sergey S. Goncharov. Autostability of models and abelian groups. *Algebra i Logika*, 19(1):23–44, 132, 1980.

[Gor70] Carl E. Gordon. Comparisons between some generalizations of recursion theory. *Compositio Math.*, 22:333–346, 1970.

[Har78] Leo Harrington. Analytic determinacy and $0^\sharp$. *J. Symbolic Logic*, 43(4):685–693, 1978.

[HKSS02] Denis R. Hirschfeldt, Bakhadyr Khoussainov, Richard A. Shore, and Arkadii M. Slinko. Degree spectra and computable dimensions in algebraic structures. *Ann. Pure Appl. Logic*, 115(1-3):71–113, 2002.

[HLZ99] Bernhard Herwig, Steffen Lempp, and Martin Ziegler. Constructive models of uncountably categorical theories. *Proc. Amer. Math. Soc.*, 127(12):3711–3719, 1999.

[HM] Kenneth Harris and A. Montalbán. Boolean algebra approximations. To appear in the Transactions of the AMS.

[HM12] Kenneth Harris and Antonio Montalbán. On the n-back-and-forth types of Boolean algebras. *Trans. Amer. Math. Soc.*, 364(2):827–866, 2012.

[HS07] Denis R. Hirschfeldt and Richard A. Shore. Combinatorial principles weaker than Ramsey's theorem for pairs. *J. Symbolic Logic*, 72(1):171–206, 2007.

[HTMM] M. Harrison-Trainor, R. Miller, and A. Montalbán. Generic functors and infinitary interpretations. In preparation.

[HTMMM] M. Harrison-Trainor, A. Melnikov, R. Miller, and A. Montalbán. Computable functors and effective interpretability. submitted for publication.

[Joc80] Carl G. Jockusch, Jr. Degrees of generic sets. In *Recursion theory: its generalisation and applications (Proc. Logic Colloq., Univ. Leeds, Leeds, 1979)*, volume 45 of *London Math. Soc. Lecture Note Ser.*, pages 110–139. Cambridge Univ. Press, Cambridge-New York, 1980.

[Kal08] I. Sh. Kalimullin. Almost computably enumerable families of sets. *Mat. Sb.*, 199(10):33–40, 2008.

[Kal09] I.Sh. Kalimullin. Uniform reducibility of representability problems for algebraic structures. *Sibirskii Matematicheskii Zhurnal*, 50(2):334–343, 2009.

[Kni86] Julia F. Knight. Degrees coded in jumps of orderings. *J. Symbolic Logic*, 51(4):1034–1042, 1986.

[Kni98] J. F. Knight. Degrees of models. In *Handbook of recursive mathematics, Vol. 1*, volume 138 of *Stud. Logic Found. Math.*, pages 289–309. North-Holland, Amsterdam, 1998.

[KP54] S.C. Kleene and E.L. Post. The upper semi-lattice of the degrees of recursive unsolvability. *Annals of Mathematics*, 59:379–407, 1954.

[KS98] Bakhadyr Khoussainov and Richard A. Shore. Computable isomorphisms, degree spectra of relations, and Scott families. *Ann. Pure Appl. Logic*, 93(1-3):153–193, 1998. Computability theory.

[KSS07] Bakhadyr Khoussainov, Pavel Semukhin, and Frank Stephan. Applications of Kolmogorov complexity to computable model theory. *J. Symbolic Logic*, 72(3):1041–1054, 2007.

[Kud96] O. V. Kudinov. An autostable 1-decidable model without a computable Scott family of $\exists$-formulas. *Algebra i Logika*, 35(4):458–467, 498, 1996.

[Lac73] A. H. Lachlan. The priority method for the construction of recursively enumerable sets. In *Cambridge Summer School in Mathematical Logic (Cambridge, 1971)*, pages 299–310. Lecture Notes in Math., Vol. 337. Springer, Berlin, 1973.

[LMMS05] Steffen Lempp, Charles McCoy, Russell Miller, and Reed Solomon. Computable categoricity of trees of finite height. *J. Symbolic Logic*, 70(1):151–215, 2005.

[LR78] Peter E. La Roche. *Contributions to Recursive Algebra*. ProQuest LLC, Ann Arbor, MI, 1978. Thesis (Ph.D.)–Cornell University.

[Mal62] Anatolii I. Mal'cev. On recursive Abelian groups. *Dokl. Akad. Nauk SSSR*, 146:1009–1012, 1962.

[Mar68] Donald A. Martin. The axiom of determinateness and reduction principles in the analytical hierarchy. *Bull. Amer. Math. Soc.*, 74:687–689, 1968.

[Mar75] Donald A. Martin. Borel determinacy. *Ann. of Math. (2)*, 102(2):363–371, 1975.

[Med55] Yu. T. Medvedev. Degrees of difficulty of the mass problem. *Dokl. Akad. Nauk SSSR (N.S.)*, 104:501–504, 1955.

[MM] A. Melnikov and A. Montalbán. Computable Polish group actions. Submitted for publication.

[Mona] A. Montalbán. Effectively existentially-atomic structures. Submitted for publication.

[Monb] Antonio Montalbán. Analytic equivalence relations satisfying hyperarithmetic-is-recursive. Submitted for publication.

[Monc] Antonio Montalbán. Classes of structures with no intermediate isomorphism problems. Submitted for publication.

[Mond] Antonio Montalbán. Computability theoretic classifications for classes of structures. Submitted for publication.

[Mone] Antonio Montalbán. Priority arguments via true stages. Submitted for publication.

[Mon09] Antonio Montalbán. Notes on the jump of a structure. *Mathematical Theory and Computational Practice*, pages 372–378, 2009.

[Mon10] Antonio Montalbán. Counting the back-and-forth types. *Journal of Logic and Computability*, page doi: 10.1093/logcom/exq048, 2010.

[Mon12] Antonio Montalbán. Rice sequences of relations. *Philosophical Transactions of the Royal Society A*, 370:3464–3487, 2012.

[Mon13a] Antonio Montalbán. A computability theoretic equivalent to Vaught's conjecture. *Adv. Math.*, 235:56–73, 2013.

[Mon13b] Antonio Montalbán. Copyable structures. *Journal of Symbolic Logic*, 78(4):1025–1346, 2013.

[Mon13c] Antonio Montalbán. A fixed point for the jump operator on structures. *Journal of Symbolic Logic*, 78(2):425–438, 2013.

[Mor04] A. S. Morozov. On the relation of Σ-reducibility between admissible sets. *Sibirsk. Mat. Zh.*, 45(3):634–652, 2004.

[Mos69] Yiannis N. Moschovakis. Abstract first order computability. I, II. *Trans. Amer. Math. Soc.*, 138:427–464, 1969.

[MPSS] R. Miller, B. Poonen, H. Schoutens, and A. Shlapentokh. A computable functor from graphs to fields. To appear.

[Muc56] A. A. Muchnik. On the unsolvability of the problem of reducibility in the theory of algorithms. *Dokl. Akad. Nauk SSSR, N.S.*, 108:194–197, 1956.

[Muč63] A. A. Mučnik. On strong and weak reducibility of algorithmic problems. *Sibirsk. Mat. Ž.*, 4:1328–1341, 1963.

[Nie09] André Nies. *Computability and randomness*, volume 51 of *Oxford Logic Guides*. Oxford University Press, Oxford, 2009.

[Nur74] A. T. Nurtazin. Strong and weak constructivizations, and enumerable families. *Algebra i Logika*, 13:311–323, 364, 1974.

[Pos44] Emil L. Post. Recursively enumerable sets of positive integers and their decision problems. *Bull. Amer. Math. Soc.*, 50:284–316, 1944.

[Pou72] Maurice Pouzet. Modèle universel d'une théorie n-complète: Modèle uniformément préhomogène. *C. R. Acad. Sci. Paris Sér. A-B*, 274:A695–A698, 1972.

[Puz09] V. G. Puzarenko. On a certain reducibility on admissible sets. *Sibirsk. Mat. Zh.*, 50(2):415–429, 2009.

[Puz11] Vadim Puzarenko. Fixed points of the jump operator. *Algebra and Logic*, 5, 2011. to appear.

[Rem81] J. B. Remmel. Recursively categorical linear orderings. *Proc. Amer. Math. Soc.*, 83(2):387–391, 1981.

[Ric77] Linda Richter. *Degrees of unsolvability of models*. PhD thesis, University of Illinois at Urbana-Champaign, 1977.

[Ric81] Linda Jean Richter. Degrees of structures. *J. Symbolic Logic*, 46(4):723–731, 1981.

[Sel71] Alan L. Selman. Arithmetical reducibilities. I. *Z. Math. Logik Grundlagen Math.*, 17:335–350, 1971.

[Sim76] H. Simmons. Large and small existentially closed structures. *J. Symbolic Logic*, 41(2):379–390, 1976.

[Sla98] Theodore A. Slaman. Relative to any nonrecursive set. *Proc. Amer. Math. Soc.*, 126(7):2117–2122, 1998.

[Smi81] Rick L. Smith. Two theorems on autostability in p-groups. In *Logic Year 1979–80 (Proc. Seminars and Conf. Math. Logic, Univ. Connecticut, Storrs, Conn., 1979/80)*, volume 859 of *Lecture Notes in Math.*, pages 302–311. Springer, Berlin, 1981.

[Soa87] Robert I. Soare. *Recursively enumerable sets and degrees*. Perspectives in Mathematical Logic. Springer-Verlag, Berlin, 1987. A study of computable functions and computably generated sets.

[Sos04] Ivan N. Soskov. Degree spectra and co-spectra of structures. *Annuaire Univ. Sofia Fac. Math. Inform.*, 96:45–68, 2004.

[Sos07] Alexandra Soskova. A jump inversion theorem for the degree spectra. In *Proceeding of CiE 2007*, volume 4497 of *Lecture Notes in Comp. Sci.*, pages 716–726. Springer-Verlag, 2007.

[SS09] Alexandra A. Soskova and Ivan N. Soskov. A jump inversion theorem for the degree spectra. *J. Logic Comput.*, 19(1):199–215, 2009.

[Ste13] Rebecca M. Steiner. Effective algebraicity. *Arch. Math. Logic*, 52(1-2):91–112, 2013.

[Stu] A. I. Stukachev. Effective model theory via the Σ-definability approach. To appear in the Proccedings of EMU 2008, Lecture Notes in Logic, vol 41.

[Stu07] A. I. Stukachev. Degrees of presentability of models. I. *Algebra Logika*, 46(6):763–788, 793–794, 2007.

[Stu09] A. I. Stukachev. A jump inversion theorem for semilattices of Σ-degrees. *Sib. Èlektron. Mat. Izv.*, 6:182–190, 2009.

[Stu10] A. I. Stukachev. A jump inversion theorem for the semilattices of Sigma-degrees [translation of mr2586684]. *Siberian Adv. Math.*, 20(1):68–74, 2010.

[Vaĭ89] Rimantas Vaĭtsenavichyus. Inner-resolvent feasible sets. *Mat. Logika Primenen.*, 1(6):9–20, 1989.

[Vat11] Stefan Vatev. Conservative extensions of abstract structures. In Benedikt Löwe, Dag Normann, Ivan N. Soskov, and Alexandra A. Soskova, editors, *CiE*, volume 6735 of *Lecture Notes in Computer Science*, pages 300–309. Springer, 2011.

[Ven92] Yu. G. Ventsov. The effective choice problem for relations and reducibilities in classes of constructive and positive models. *Algebra i Logika*, 31(2):101–118, 220, 1992.

[Weh98] Stephan Wehner. Enumerations, countable structures and Turing degrees. *Proc. Amer. Math. Soc.*, 126(7):2131–2139, 1998.