COPYABLE STRUCTURES

ANTONIO MONTALBÁN

Abstract. We introduce the notions of copyable and diagonalizable classes of structures. We then show how these notions are connected to two other notions that had already been studied for some particular classes of structures, namely the listability property and the low property.

The main result of this paper is the characterizations of the classes of structures with the low property, that is, the classes whose low members all have computable copies. We characterize these classes as the ones whose structural jumps are listable.

1. Introduction

This paper is part of the ongoing project of understanding the computability theoretic properties of mathematical structures. We are particularly interested in the following two questions: what are conditions that guarantee that a given structure can be represented computably, and how difficult is it to computably enumerate all the computable structures in a given class? We will study an instance of the former question by looking at the "low property" (see Definition 1.3 below). We will study the latter question by looking at the "listability property" (see Definition 1.1 below). The actual goal of this paper is, however, to introduce two new notions, the one of *copyability* and the one of *diagonalizability*, which we will use to better understand the former two.

The game. To each class of structures $\mathbb{K}$ we will associate a game $\mathsf{G}(\mathbb{K})$. In this game, players C and D build one structure in $\mathbb{K}$ each. The goal of C is to get both structures to be isomorphic (to copy), and the goal of D is to get the structures to be different (to diagonalize). Depending on the structural properties of the class $\mathbb{K}$, one or the other player will have a winning strategy, and we will be able to use that strategy to obtain computability theoretic properties of $\mathbb{K}$. If C has a computable winning strategy, we say $\mathbb{K}$ is *copyable*, and if D has a computable winning strategy, we say $\mathbb{K}$ is *diagonalizable*. (See Section 2 for the formal definitions.) The idea of copying or diagonalizing a structure is behind many proofs already in the literature, and, in many of these proofs, our game captures the essential combinatorial aspects behind the proofs. The definition of this game is new, but all we are doing is putting old ideas in a single concrete general framework so that we we can prove general results about them. As an example, we analyze the class of linear orderings in detail in Section 7.

Listable classes. In Lemma 2.5 we show that if a class is copyable, it is listable.

[0] Saved: August 17, 2012 - Submitted
Compiled: August 17, 2012

2000 *Mathematics Subject Classification.* 03D80.

The author was partially supported by NSF grant DMS-0901169 and the Packard Fellowship. The author would like to thank Asher Kach for letting him include the proof Lemmas 7.3 and 7.4 which were done in joint work. The author would also like to thank Victor Ocasio, Jonathan Stephenson and Steven VanDendriessche for proof reading this paper.

Definition 1.1. A class $\mathbb{K}$ is *listable* if there exists a Turing functional which, for every oracle X, produces an X-computable sequence of structures listing all the X-computable structures in $\mathbb{K}$ (allowing repetitions).

Again, even if this definition is new as is, it is not a new notion. Nurtazin [Nur74], almost four decades ago, gave a sufficient condition for a class of structures to be listable which includes the classes of linear orderings, Boolean algebras, equivalence structures, Abelian p-groups, and algebraic fields of characteristic p. Nurtazin's result says that if there exists a (universal) computable structure in the class such that any other structure can be embedded into it, and such that any subset of that structure generates a structure in the class, then the class is listable (see [GN02, Theorem 5.1]). Goncharov and Knight [GN02, Section 5] consider a similar notion as their "third approach" to defining the notion of a class having a computable characterization.

Neither Nurtazin's condition, nor copyability is a necessary condition for a class to be listable. However, using a modified version of our game $\mathsf{G}(\mathbb{K})$ we obtain a characterization of listable classes. In Section 3 we will define the game $\mathsf{G}^\infty(\mathbb{K})$, where C is allowed to build infinitely many structures, and needs to get just one of them to be isomorphic to the one played by D. Our first theorem characterizes listable classes as those where C has a winning strategy in the game $\mathsf{G}^\infty(\mathbb{K})$, (which we call ∞-copyable classes).

Theorem 1.2. *A class $\mathbb{K}$ of infinite structures is listable if and only if it is ∞-copyable.*

One might ask why it is that we are introducing the notion of ∞-copyability if it is equivalent to the notion of listability, which is much more natural and simpler to define. The reason is that many times when we want to prove that a class is listable, or not listable, the argument goes, essentially, by showing that the class is either ∞-copyable or ∞-diagonalizable.

The low property. The following is the relativized version of what it is usually called the low property.

Definition 1.3. We say that $\mathbb{K}$ has the *low property* if, for every $X, Y \in 2^\omega$ with $X' \equiv_T Y'$, and every structure $\mathcal{A} \in \mathbb{K}$, we have that

$$\mathcal{A} \text{ has an } X\text{-computable copy} \iff \mathcal{A} \text{ has a } Y\text{-computable copy.}$$

This property has been studied in various examples. Most famously, the class of linear ordering has been shown not to have the low property by Jockusch and Soare [JS91], and the class of Boolean algebras to have the low property by Downey and Jockusch [DJ94]. Moreover, the class of Boolean algebras has the low_4 property as shown by Knight and Stob [KS00]. The question of whether Boolean algebras have the low_n property has been open for almost twenty years. (As expected, the low_n property is defined exactly as in 1.3, but with the assumption that $X^{(n)} \equiv_T Y^{(n)}$.) Also, the class of scattered linear orderings has the low_2 property as was recently claimed by Frolov (unpublished), and the class of linear orderings with finitely many descending sequences has the low_n property for all $n \in \omega$ as shown by Kach and the author [KM11].

A reasonable expectation some researcher had was that when a class has the low property, it has it for a reason (since the low property is such a natural computational property). What that reason would look like has been unknown for a couple decades. The main result of this paper is to give a somewhat satisfactory structural characterization of the classes of structures with the low property in terms of the notions of structural jump and listability.

The essential combinatorial properties behind the analysis of the low properties in the known examples can be captured by the game $\mathsf{G}^\infty(\mathbb{K}_{(1)})$ (where $\mathbb{K}_{(1)}$ is the class of structural jumps of the structures in $\mathbb{K}$ as defined in 4.3). This fact is captured by our main theorem

below, which characterizes the classes $\mathbb{K}$ with the low property assuming some effectiveness condition on $\mathbb{K}$. This effectiveness condition, namely that $\mathbb{K}$ has a 1-back-and-forth structure, is described in Section 4. This condition applies to the classes of linear orderings, Boolean algebras, etc.

Theorem 1.4. *Let $\mathbb{K}$ be a class of infinite structures with a computable 1-back-and-forth structure.*

(1) *If $\mathbb{K}$ is axiomatizable by a Π^c_2-sentence and $\mathbb{K}_{(1)}$ is ∞-diagonalizable, then $\mathbb{K}$ does not have the low property.*
(2) *If $\mathbb{K}_{(1)}$ is ∞-copyable, it has the low property.*

Some of the technical tools necessary for the both parts were developed by Harris and the author [HM12, HM] for the class of Boolean algebras.

In [Mon09, Theorem 3.5] the author showed that the low property is equivalent to saying that, for every $X \in 2^\omega$ and every $\mathcal{A} \in \mathbb{K}$,

$$\mathcal{A} \text{ has an } X\text{-computable copy} \iff \mathcal{A}' \text{ has an } X'\text{-computable copy},$$

where $\mathcal{A}'$ is as defined in [Mon09, Mona]. It follows that $\mathbb{K}$ has the low$_n$ property if and only if, for each $m < n$, the class of mth jumps of the structures in $\mathbb{K}$ has the low property. (See [Mona] for the definition of the mth jump of a structure.) This is why it is enough to restrict our attention to the low property, and not the low$_n$ property.

Adding guesses to the game. In Section 6 we introduce the game $\mathsf{G}^\alpha(\mathbb{K})$, where α is a computable ordinal. These games are in between $\mathsf{G}(\mathbb{K})$ and $\mathsf{G}^\infty(\mathbb{K})$, and are usually the games that are played in "practice." The key lemma behind the proof by Harris and the author [HM12, HM] that there is a low$_5$ Boolean algebra not $0^{(7)}$-isomorphic to a computable one says, essentially, that $\mathbb{BA}^{(5)}$, the class of 5-jumps of Boolean algebras, is 2-diagonalizable. Whether this class is 3-copyable, or even ∞-copyable is unknown.

In the last section we show that the class of linear orderings is 2-diagonalizable and 4-copyable. The former result is due to Kach and the author (unpublished), but the main ideas already appeared in Jockusch and Soare [JS91] and also in R. Miller [Mil01]. We do not know whether linear orderings are 3-copyable or 3-diagonalizable.

2. The game

Let $\mathbb{K}$ be a class of infinite $\mathcal{L}$-structures, where $\mathcal{L}$ is a relational language. By an *approximation* to a structure $\mathcal{C}$ we mean a sequence $\{\mathcal{C}[s] : s \in \omega\}$ of finite $\mathcal{L} \upharpoonright k_s$-structures for some $k_s \in \omega$, (where $\mathcal{L} \upharpoonright k$ denotes the first k symbols in $\mathcal{L}$), such that for every s, $k_s \leq k_{s+1}$, $\mathcal{C}[s]$ is included in $\mathcal{C}[s+1]$ as an $\mathcal{L} \upharpoonright k_s$-structure, and

$$\mathcal{C} = \bigcup_s \mathcal{C}[s].$$

(This is equivalent to consider finite approximations to the atomic diagram of $\mathcal{C}$.)

The *game* $\mathsf{G}(\mathbb{K})$ has two players, C and D, each one trying to build an approximation to a structure in $\mathbb{K}$. The goal of C is to get both structures to be isomorphic, and the goal of D is to get the structures to be different. Players play alternatively and who starts is irrelevant for the game (for concreteness, let us say that C starts the game). On the sth move, C plays a finite $\mathcal{L} \upharpoonright k_s$-structure $\mathcal{C}[s]$ and D plays a finite $\mathcal{L} \upharpoonright \ell_s$-structure $\mathcal{D}[s]$, for some $k_s, \ell_s \in \omega$. (We allow the *empty structure* as a structure.) At stage $s+1$, player C must make sure that $k_s \leq k_{s+1}$ and that $\mathcal{C}[s] \subseteq \mathcal{C}[s+1]$ as $\mathcal{L} \upharpoonright k_s$-structures, as otherwise he immediately looses. Analogously, player D must make sure that $\ell_s \leq \ell_{s+1}$, and $\mathcal{D}[s] \subseteq \mathcal{D}[s+1]$ as $\mathcal{L} \upharpoonright \ell_s$-structures, as otherwise he immediately loses. At the end of stages we let $\mathcal{C} = \bigcup_s \mathcal{C}[s]$ and $\mathcal{D} = \bigcup_s \mathcal{D}[s]$.

Player D		$\mathcal{D}[0]$	$\subseteq$	$\mathcal{D}[1]$	$\subseteq$	$\mathcal{D}[2]$	$\subseteq$	$\cdots$	$\mathcal{D} = \bigcup_s \mathcal{D}[s]$
Player C	$\mathcal{C}[0]$	$\subseteq$	$\mathcal{C}[1]$	$\subseteq$	$\mathcal{C}[2]$	$\subseteq$	$\mathcal{C}[3]$	$\cdots$	$\mathcal{C} = \bigcup_s \mathcal{C}[s]$

We then decide who wins as follows.

(1) If $\mathcal{C} \notin \mathbb{K}$, then D wins.
(2) If $\mathcal{C} \in \mathbb{K}$, but $\mathcal{D} \notin \mathbb{K}$, then C wins.
(3) If $\mathcal{C}, \mathcal{D} \in \mathbb{K}$ are isomorphic then C wins.
(4) If $\mathcal{C}, \mathcal{D} \in \mathbb{K}$ are not isomorphic then D wins.

(We note that, for $\mathcal{C}$ to be an $\mathcal{L}$-structure in $\mathbb{K}$, $\lim_s k_s$ must be equal to the size of $\mathcal{L}$. If $\mathcal{L}$ is finite, without loss of generality, we might assume that k_s is always equal to the size of $\mathcal{L}$.)

An important remark is that the players are allowed to *pass*, in the sense that they can play $\mathcal{C}[s+1] = \mathcal{C}[s]$ or $\mathcal{D}[s+1] = \mathcal{D}[s]$. However, C is forced to build a structure in $\mathbb{K}$ at the end, and all structures in $\mathbb{K}$ are infinite. Thus, independently of what D does, C is obligated to extend his finite structures infinitely often, because if both players stop extending their structures, the one who loses is C. In other words, D can wait until he sees enough of $\mathcal{C}$ to extend his structure one more bit. This is a key feature of the game $\mathsf{G}(\mathbb{K})$, and, whether D can make use of this advantage or not will depend on the structural properties of $\mathbb{K}$. An equivalent formulation of the game would be to forbid C to pass (i.e., he has to build proper extensions at each step), while D is allowed to pass so long as he extends his structure infinitely often.

Definition 2.1. We say that $\mathbb{K}$ is *copyable* if C has a computable strategy in the game $\mathsf{G}(\mathbb{K})$. We say that $\mathbb{K}$ is *diagonalizable* if D has a computable strategy in the game $\mathsf{G}(\mathbb{K})$.

Given an oracle X, we can define the notions of *X-copyable* and *X-diagonalizable* by requiring the strategies above to be X-computable. Of course, if $\mathbb{K}$ is X-copyable for some X, it cannot be Y-diagonalizable for any Y. Notice that if we have enough determinacy (for instance, analytic determinacy when $\mathbb{K}$ is borel), there must exist an $X \in 2^\omega$ such that $\mathbb{K}$ is either X-copyable or X-diagonalizable. We expect that whenever $\mathbb{K}$ is natural enough, this oracle X should be 0 and no appeal to determinacy should be necessary.

Example 2.2. The class of infinite linear orderings is diagonalizable. (We include a proof below in Section 7 due to Kach and the author.) The ideas in this proof are not really new, as they were already used by Jockusch and Soare [JS91] when they build a low linear ordering without a computable copy, and by R. Miller [Mil01] when he built a non-computable linear ordering computable from all non-computable Δ^0_2 sets. The notion of diagonalizable essentially isolates the key combinatorial argument needed to satisfy a single requirement in their proofs.

Example 2.3. The class of Boolean algebras, with a predicate for *atom*, and with infinitely many atoms, is copyable. The ideas in this proof are also know as they were present in Downey and Jockusch's [DJ94] proof that every low Boolean algebras has a computable copy.

2.1. **Listable classes.**

Recall from 1.1 the definition of a listable class of structures. We show in this section that if a class is copyable, it is listable.

Example 2.4. The class of linear orderings is listable. This is because a linear ordering $\mathcal{L}$ has an X-computable copy if and only if it is isomorphic to an X-c.e. subset of the rationals. Since X can uniformly lists all X-c.e. sets, it can lists all X-computable linear orderings. Notice that this includes all finite linear orderings. The class of infinite linear orderings is also listable, but showing it requires more effort. It will follow from Lemma 7.2.

Lemma 2.5. *If $\mathbb{K}$ is copyable, it is listable.*

Proof. Suppose we have a computable strategy for C in the game $\mathsf{G}(\mathbb{K})$. Fix an oracle X; we will build a list $\{\mathcal{C}_e : e \in \omega\}$ of all X-computable structures in $\mathbb{K}$. For each e, we will define $\mathcal{C}_e$ to be the outcome of C's strategy when playing against a player D who is playing according to $\{e\}^X$, the eth Turing function with oracle X, as explained below.

Fix $e \in \omega$. Using oracle X, we define a sequence of finite structures $\{\mathcal{D}[s] : s \in \omega\}$ and an auxiliary sequence of numbers $\{n_s : s \in \omega\}$ as follows.

- At stage $s = 0$, let $\mathcal{D}[0]$ be the empty structure on the empty language. Let $n_0 = 0$.
- At stage $s+1$, if $\{e\}^X_s(n_s)$ converges and is equal to the index of a finite structure $\mathcal{A}$ extending $\mathcal{D}[s]$, we define $\mathcal{D}[s+1] = \mathcal{A}$, and let $n_{s+1} = n_s + 1$. If, either $\{e\}^X_s(n_s)$ diverges, or converges but not to the index of a finite structure extending $\mathcal{D}[s]$, we let $\mathcal{D}[s+1] = \mathcal{D}[s]$ and $n_{s+1} = n_s$.

Let $\mathcal{C}_e$ be the structure played by C, when his following his strategy against the sequence $\{\mathcal{D}[s] : s \in \omega\}$. Since C wins, $\mathcal{C}_e$ must be a structure in $\mathbb{K}$. Since the strategy is computable and the sequence $\{\mathcal{D}[s] : s \in \omega\}$ is X-computable, $\mathcal{C}_e$ is also X-computable. When $\{e\}^X$ is actually enumerating the indices of an approximation to a structure in $\mathbb{K}$, $\mathcal{C}_e$ must be isomorphic to it. So $\{\mathcal{C}_e : e \in \omega\}$ is a listing of all X-computable structures in $\mathbb{K}$. □

The converse of this lemma is not true. But it is, if we consider the infinite version of our game.

3. The infinity game

This game is similar to the previous game except that C is allowed to build infinitely many structures rather than just one, and he only needs to get one of them to be isomorphic to the structure played by D. It is a much harder game for D, as he now must diagonalize against infinitely many structures simultaneously.

We now define the game $\mathsf{G}^\infty(\mathbb{K})$. Again, players C and D play alternatively and, again, it is irrelevant who starts. On the sth move, D plays a finite $\mathcal{L} \restriction \ell_s$-structure $\mathcal{D}[s]$ for some ℓ_s, and C plays $s+1$ many finite $\mathcal{L} \restriction k_{j,s}$-structures $\mathcal{C}^j[s-j]$ for $j = 0, ..., s$. Structures must be built in chains exactly as in the previous game (i.e., $\ell_s \leq \ell_{s+1}$, $\mathcal{D}[s] \subseteq \mathcal{D}[s+1]$, $k_{j,s} \leq k_{j,s+1}$ and $\mathcal{C}^j[s] \subseteq \mathcal{C}^j[s+1]$), and whoever does not follow this rule loses. At the end of stages we let $\mathcal{C}^j = \bigcup_s \mathcal{C}^j[s]$ and $\mathcal{D} = \bigcup_s \mathcal{D}[s]$.

Player D		$\mathcal{D}[0]$	$\subseteq$	$\mathcal{D}[1]$	$\subseteq$	$\mathcal{D}[2]$	$\cdots$	$\mathcal{D} = \bigcup_s \mathcal{D}[s]$
	$\mathcal{C}^0[0]$	$\subseteq$	$\mathcal{C}^0[1]$	$\subseteq$	$\mathcal{C}^0[2]$	$\subseteq$	$\cdots$	$\mathcal{C}^0 = \bigcup_s \mathcal{C}^0[s]$
Player C			$\mathcal{C}^1[0]$	$\subseteq$	$\mathcal{C}^1[1]$	$\subseteq$	$\cdots$	$\mathcal{C}^1 = \bigcup_s \mathcal{C}^1[s]$
					$\mathcal{C}^2[0]$	$\subseteq$	$\cdots$	$\mathcal{C}^2 = \bigcup_s \mathcal{C}^1[s]$
						$\ddots$	$\vdots$	$\vdots$

We then decide who wins the game $\mathsf{G}^\infty(\mathbb{K})$ as follows:

(1) If for some j, $\mathcal{C}^j \notin \mathbb{K}$, then D wins.
(2) If for all j, $\mathcal{C}^j \in \mathbb{K}$, but $\mathcal{D} \notin \mathbb{K}$, then C wins.
(3) If $\mathcal{D}, \mathcal{C}^0, \mathcal{C}^1, ... \in \mathbb{K}$ and, for some j, $\mathcal{D} \cong \mathcal{C}^j$, then C wins.
(4) If $\mathcal{D}, \mathcal{C}^0, \mathcal{C}^1, ... \in \mathbb{K}$ and, for all j, $\mathcal{D} \ncong \mathcal{C}^j$, then D wins.

Definition 3.1. We say that $\mathbb{K}$ is ∞-*copyable* if C has a computable strategy in the game $\mathsf{G}^\infty(\mathbb{K})$. We say that $\mathbb{K}$ is ∞-*diagonalizable* if D has a computable strategy in the game $\mathsf{G}^\infty(\mathbb{K})$.

Theorem 3.2. *A class $\mathbb{K}$ of infinite structures is listable if and only if it is ∞-copyable.*

Proof. Suppose first that $\mathbb{K}$ is ∞-copyable. Then, essentially by the same proof as that of Lemma 2.5, we can build an X-computable sequence of structures $\{\mathcal{C}_e^j : e, j \in \omega\}$ all in $\mathbb{K}$ and such that if $\{e\}^X$ is a structure in $\mathbb{K}$, then for some j, $\mathcal{C}_e^j$ is isomorphic to it.

Suppose now that $\mathbb{K}$ is listable; we need to define a strategy for C. Let X be the sequence of indices of the finite structures played by D. We let the C-strategy play the X-computable list of all X-computable structures in $\mathbb{K}$ as response. Since $\mathcal{D}$ is computable in X, it will be isomorphic to one of the structures played by C. □

4. The back-and-forth structure

The connections between copyability and the low property work under some effectiveness conditions on the class $\mathbb{K}$, namely that it has a computable 1-back-and-forth structure. We do not know what happens when this effectiveness condition is not present. The α-back-and-forth structures were developed in [Mon10, Monb] generalizing ideas from Harris and the author [HM12] about Boolean algebras. In this paper we only need to look at the case $\alpha = 1$. We will review all the necessary background in this simplified scenario of $\alpha = 1$.

Definition 4.1. We say that $\mathbb{K}$ has a *computable 1-back-and-forth structure* if there is a list $\{\sigma_i : i \in \omega\}$ of all the (finitary) Π_1-$\mathcal{L}$-types realized in $\mathbb{K}$ such that the following decision procedures are computable:

- given i, and a Π_1 formula, deciding whether the formula is in σ_i;
- given i, deciding what are the free variables in the type σ_i;
- given i, j, and a subset of the free variables of σ_i, deciding if σ_j is the restriction of σ_i to those variables;
- given i, j, deciding if $\sigma_i \subseteq \sigma_j$ as sets of Π_1 formulas.

We note that it is only possible to have a computable 1-back-and-forth structure if there are only countably many Π_1-$\mathcal{L}$-types realized in $\mathbb{K}$, which is the case in some natural classes of structures and not in others. Among the natural classes of structures which do realize only countably many Π_1-$\mathcal{L}$-types, they usually have a computable 1-back-and-forth structure (see [Mon10, Section 4 on examples]).

For the rest of this section fix a class $\mathbb{K}$ with a computable 1-back-and-forth structure.

Notation 4.2. We use $\mathbf{bf}_1$ to denote the set of all Π_1-types realized in $\mathbb{K}$. That is

$$\mathbf{bf}_1 = \{\Pi_1\text{-tp}_{\mathcal{A}}(\overline{a}) : \overline{a} \in \mathcal{A}^{<\omega}, \mathcal{A} \in \mathbb{K}\},$$

where $\Pi_1\text{-tp}_{\mathcal{A}}(\overline{a}) = \{\psi(\overline{x}) : \psi(\overline{x}) \text{ is a } \Pi_1\text{-}\mathcal{L}\text{-formula}, \mathcal{A} \models \psi(\overline{a})\}$. Given $\sigma \in \mathbf{bf}_1$, we let $|\sigma|$ be the number of free variables in σ, which we assume are $\overline{x} = x_1, ..., x_{|\sigma|}$. Given $\sigma, \tau \in \mathbf{bf}_1$ with $|\sigma| = |\tau|$, we let $\sigma \leq_1 \tau$ if $\sigma \subseteq \tau$ as sets of Π_1 formulas.

For each $\sigma \in \mathbf{bf}_1$ on the variables $\overline{x} = (x_1, ...x_k)$, let

$$\boldsymbol{\varphi}_\sigma(\overline{x}) \quad \equiv \quad \bigwedge\{\psi(\overline{x}) : \psi(\overline{x}) \in \sigma\}.$$

So, we have that $\mathcal{A} \models \boldsymbol{\varphi}_\sigma(\overline{a})$ if and only if $\sigma \subseteq \Pi_1\text{-tp}_{\mathcal{A}}(\overline{a})$ (or, equivalently, $\sigma \leq_1 \Pi_1\text{-tp}_{\mathcal{A}}(\overline{a})$). It was shown in [Mon10] that these formulas form a complete set of Π_1^{c}-relations in the sense that every Σ_2^{c}-$\mathcal{L}$-formula is equivalent (on all the structures in $\mathbb{K}$) to a $\Sigma_1^{\mathsf{c},0'}$ formula in the language $\mathcal{L} \cup \{\boldsymbol{\varphi}_\sigma : \sigma \in \mathbf{bf}_1\}$, and this equivalent formula can be found uniformly. (Where Π_1^{c} and Σ_2^{c} are the sets of *computable infinitary* Π_1 and Σ_2 formulas, and $\Sigma_1^{\mathsf{c},0'}$ is the set of $0'$-computable infinitary Σ_1 formulas.)

Definition 4.3. We define an extended language

$$\mathcal{L}_1 = \mathcal{L} \cup \{\boldsymbol{\varphi}_\sigma : \sigma \in \mathbf{bf}_1\}.$$

Given $\mathcal{A} \in \mathbb{K}$, we call the $\mathcal{L}_1$-structure

$$\mathcal{A}_{(1)} = (\mathcal{A}; \boldsymbol{\varphi}_\sigma^{\mathcal{A}}, \sigma \in \mathbf{bf}_1),$$

where $\boldsymbol{\varphi}_\sigma^{\mathcal{A}} = \{\overline{a} : \mathcal{A} \models \boldsymbol{\varphi}_\sigma(\overline{a})\}$, *the structural jump of* $\mathcal{A}$. We use $\mathbb{K}_{(1)}$ to denote the class of structural jumps of structures in $\mathbb{K}$:

$$\mathbb{K}_{(1)} = \{\mathcal{A}_{(1)} : \mathcal{A} \in \mathbb{K}\}.$$

If $\mathcal{A}$ is a linear ordering one can show that $\mathcal{A}_{(1)}$ is, in a certain sense, equivalent to $(\mathcal{A}, Adj(x,y))$, where $Adj(x,y)$ is the adjacency relation. These structures are equivalent in the sense that they can compute one another (even without changing the domain). The same is true for $(\mathcal{A}, atom(x))$ when $\mathcal{A}$ is a Boolean algebra and $atom(x)$ is the atom relation. So, the results that we state about $\mathcal{A}_{(1)}$ below are also true for these equivalent structures.

For the proof that copyable classes have the low property, we need to introduce some new notation to deal with approximation to $\mathcal{L}_1$-structures. The notion of 1-approximation given below was introduced by Harris and the author for the class of Boolean algebras in [HM].

Definition 4.4. A *finite labeled structure* is a pair $\hat{\mathcal{C}} = (\mathcal{C}, \mathfrak{t}^{\mathcal{C}})$ where $\mathcal{C}$ is a finite $\mathcal{L}$-structure and $\mathfrak{t}^{\mathcal{C}} \colon C^{<\omega} \to \mathbf{bf}_1$ such that, if $\overline{a}, \overline{b} \in \mathcal{C}$,

- the atomic diagram of $\overline{a}$ is as given by the atomic formulas in $\mathfrak{t}^{\mathcal{C}}(\overline{a})$, and
- $\mathfrak{t}^{\mathcal{C}}(\overline{a})$ is the restriction of $\mathfrak{t}^{\mathcal{C}}(\overline{a}, \overline{b})$ to the first $|\overline{a}|$-many variables.

Notice that in a finite labeled structure $(\mathcal{C}_0, \mathfrak{t}^{\mathcal{C}})$, if $\mathfrak{t}^{\mathcal{C}}(\overline{a}) = \sigma$, ψ is Π_1 and $\psi \notin \sigma$, then it does not need to be the case that $\mathcal{C}_0 \not\models \psi(\overline{a})$. The idea is that we are viewing $\mathcal{C}_0$ as a substructure of a structure $\mathcal{C}$ where $\overline{a}$ has Π_1-type $\mathfrak{t}^{\mathcal{C}}(\overline{a})$.

Notice also that the map $\mathfrak{t}^{\mathcal{C}}$ only needs to be defined on the tuples whose elements are all different, so $\mathfrak{t}^{\mathcal{C}}$ is really a finite object. We remark that, since all Π_1-types are computable and we have a list of them, the elements of $\mathbf{bf}_1$ should be thought of as finite objects, namely the indices i for the Π_1-types σ_i.

Definition 4.5. Given $\hat{\mathcal{C}}_0 = (\mathcal{C}_0, \mathfrak{t}_0)$ and an infinite structure $\mathcal{C} \in \mathbb{K}$, with $\mathcal{C}_0 \subseteq \mathcal{C}$, we say that $\hat{\mathcal{C}}_0$ *is correct within* $\mathcal{C}$ if for all $\overline{a} \in \mathcal{C}_0^{<\omega}$, $\mathfrak{t}_0(\overline{a}) = \Pi_1\text{-tp}_{\mathcal{C}}(\overline{a})$.

Given finite labeled structures $\hat{\mathcal{C}}_0 = (\mathcal{C}_0, \mathfrak{t}_0)$ and $\hat{\mathcal{C}}_1 = (\mathcal{C}_1, \mathfrak{t}_1)$, we say that

$$\hat{\mathcal{C}}_0 \leq_0 \hat{\mathcal{C}}_1$$

if $\mathcal{C}_0 \subseteq \mathcal{C}_1$. We say that

$$\hat{\mathcal{C}}_0 \leq_1 \hat{\mathcal{C}}_1$$

if $\mathcal{C}_0 \subseteq \mathcal{C}_1$ and for all $\overline{a} \in \mathcal{C}_0^{<\omega}$, $\mathfrak{t}_0(\overline{a}) \leq_1 \mathfrak{t}_1(\overline{a})$.

Given a subset $E \subseteq \mathcal{C}_0$ with $E \subseteq \mathcal{C}_1$ too, we say that

$$\hat{\mathcal{C}}_0 \equiv_1^E \hat{\mathcal{C}}_1$$

if for all $\overline{a} \in E^{<\omega}$, $\mathfrak{t}_0(\overline{a}) = \mathfrak{t}_1(\overline{a})$.

The following lemma gives what is known as the "back-and-forth" definition of the relation $\leq_1$.

Lemma 4.6. *Suppose that* $\hat{\mathcal{C}}_0 = (\mathcal{C}_0, \mathfrak{t}_0)$ *is correct within* $\mathcal{C}$. *Let* $(\mathcal{C}_1, \mathfrak{t}_1)$ *be a finite labeled structure such that* $(\mathcal{C}_0, \mathfrak{t}_0) \leq_1 (\mathcal{C}_1, \mathfrak{t}_1)$. *Then there exists an embedding of* $\mathcal{C}_1$ *into* $\mathcal{C}$ *fixing* $\mathcal{C}_0$.

Proof. Let $\overline{c}$ be the tuple of $\mathcal{C}$ that lists the elements of $\mathcal{C}_0$. Let $\tau = \Pi_1\text{-tp}_{\mathcal{C}}(\overline{c}) = \mathfrak{t}_0(\overline{c})$. Since $(\mathcal{C}_0, \mathfrak{t}_0) \leq_1 (\mathcal{C}_1, \mathfrak{t}_1)$, $\tau \leq_1 \mathfrak{t}_1(\overline{c})$. The Π_1 formula saying that there is no extension of $\overline{c}$ with the same atomic diagram as $\mathcal{C}_1$ is not part of $\mathfrak{t}_1(\overline{c})$, and hence not part of $\mathfrak{t}_0(\overline{c}) = \Pi_1\text{-tp}_{\mathcal{C}}(\overline{c})$ either. Therefore, there is an extension of $\overline{c}$ in $\mathcal{C}$ with the same atomic diagram as $\mathcal{C}_1$. $\square$

We notice that for $\mathcal{C} \in \mathbb{K}$, $\mathcal{C}_{(1)}$ might not be able to compute the function that maps a tuple $\overline{a}$ to $\sigma = \Pi_1\text{-tp}_{\mathcal{C}}(\overline{a}) \in \mathbf{bf}_1$. It can, however, approximate this function from below as in the following definition.

Definition 4.7. Given $\mathcal{C} \in \mathbb{K}$, a *1-approximation* to $\mathcal{C}$ is a sequence of finite labeled structures $\hat{\mathcal{C}}[s] = (\mathcal{C}[s], \mathfrak{t}[s])$ such that $\mathcal{C} = \bigcup_s \mathcal{C}[s]$ and for every s

- $\hat{\mathcal{C}}[s] \leq_1 \hat{\mathcal{C}}[s+1]$,
- for every $\overline{a} \in \mathcal{C}[s]^{<\omega}$, $\exists s_0 \forall s \geq s_0$, $\mathfrak{t}[s](\overline{a}) = \Pi_1\text{-tp}_{\mathcal{C}}(\overline{a})$.

(We will write $\mathfrak{t}(\overline{a})[s]$ rather than $\mathfrak{t}[s](\overline{a})$.)

Lemma 4.8. *Let $\mathcal{C}$ be a structure in $\mathbb{K}$.*

1. *Uniformly from a 1-approximation to $\mathcal{C}$, we can compute an approximation to $\mathcal{C}_{(1)}$.*
2. *Uniformly from an approximation to $\mathcal{C}_{(1)}$, we can compute a 1-approximation to $\mathcal{C}$.*

Proof. Suppose first that we have a 1-approximation to $\mathcal{C}$ and we need to decide whether $\mathcal{C} \models \boldsymbol{\varphi}_\sigma(\overline{a})$ for each tuple $\overline{a}$ and each $\sigma \in \mathbf{bf}_1$. On the one hand, deciding if $\mathcal{C} \models \boldsymbol{\varphi}_\sigma(\overline{a})$ is Π^0_1 in $\mathcal{C}$, just by definition of $\boldsymbol{\varphi}_\sigma$. On the other hand, $\mathcal{C} \models \boldsymbol{\varphi}_\sigma(\overline{a})$ if, and only if, there exists s such that $\mathfrak{t}(\overline{a})[s] \geq_1 \sigma$. So, deciding $\mathcal{C} \models \boldsymbol{\varphi}_\sigma(\overline{a})$ is Δ^0_1 in the 1-approximation to $\mathcal{C}$.

For the second part, suppose we have a computable presentation of $\mathcal{C}_{(1)}$ (given from its approximation). We start by defining a computable function $\mathfrak{f} \colon \mathcal{C}^{<\omega} \times \omega \to \mathbf{bf}_1$ such that for every tuple $\overline{a} \in \mathcal{C}^{<\omega}$,

- $(\forall s)\ \mathfrak{f}(\overline{a}, s) \leq_1 \mathfrak{f}(\overline{a}, s+1)$ and
- $(\exists t)(\forall s \geq t)\ \mathfrak{f}(\overline{a}, s) = \mathfrak{t}(\overline{a})$.

Recall that $\mathfrak{t}(\overline{a})$ satisfies that for all $\tau \in \mathbf{bf}_1$ with $|\tau| = |\overline{a}|$,

$$\mathcal{C} \models \boldsymbol{\varphi}_\tau(\overline{a}) \iff \tau \leq_1 \mathfrak{t}(\overline{a}),$$

or, in other words, $\mathfrak{t}(\overline{a})$ is the $\leq_1$-largest σ such that $\mathcal{C} \models \boldsymbol{\varphi}_\sigma(\overline{a})$. To define $\mathfrak{f}(\overline{a}, s)$ we can look for the $\leq_1$ largest σ that has appeared so far with $\mathcal{C} \models \boldsymbol{\varphi}_\sigma(\overline{a})$. More concretely, at each s, look for $\sigma \in \mathbf{bf}_1$ such that $\mathcal{C} \models \boldsymbol{\varphi}_\sigma(\overline{a})$ and for all $\tau \in \mathbf{bf}_1$ with Gödel number below s and $|\tau| = |\overline{a}|$ we have that $\mathcal{C} \models \boldsymbol{\varphi}_\tau(\overline{a}) \iff \tau \leq_1 \sigma$. Such a σ always exists (namely $\sigma = \mathfrak{t}(\overline{a})$), and for all s greater than the Gödel number of $\mathfrak{t}(\overline{a})$ we have $\mathfrak{f}(\overline{a}, s) = \mathfrak{t}(\overline{a})$.

We will now use $\mathfrak{f}$ to define a 1-approximation to $\mathcal{C}$. The only reason why we have not done so already is that for a fixed s, the types $\mathfrak{f}(\overline{a}, s)$ might not be consistent with each other for the different tuples $\overline{a} \in \mathcal{C}[s]^{<\omega}$. Let $\mathcal{C}[s]$ be the substructure of $\mathcal{C}$ whose domain is the first s elements of $\mathcal{C}$, namely $c_1, ..., c_s = \overline{c}$. We now need to define $\mathfrak{t}(\overline{a})[s]$ for all tuples $\overline{a} \in \mathcal{C}[s]^{<\omega}$. Notice that is enough to define $\mathfrak{t}(c_1, ..., c_s)[s]$, and then take the restriction of this Π_1-type to its sub-tuples. Search for $t > s$ such that if we assign $\mathfrak{f}(\overline{c}, t)$ to the elements of $\mathcal{C}[s]$, then, for every $\overline{a} \in \mathcal{C}[s-1]$, if τ is a restriction of $\mathfrak{f}(\overline{c}, t)$ to $\overline{a}$, then

1. $\tau \geq_1 \mathfrak{t}(\overline{a})[s-1]$, and
2. $\tau \geq_1 \mathfrak{f}(\overline{a}, s)$.

Such a t exists, namely any t with $\mathfrak{f}(\overline{c}, t) = \mathfrak{t}(\overline{c})$. Let us define $\mathfrak{t}(\overline{c})[s] = \mathfrak{f}(\overline{c}, t)$.

For each tuple $\overline{a}$ we have that, on one hand $\mathfrak{t}(\overline{a})[s] \leq_1 \mathfrak{t}(\overline{a})$ because $\mathfrak{t}(\overline{c})[s] \leq_1 \mathfrak{t}(\overline{c})$, and, on the other hand, $\mathfrak{f}(\overline{a}, s) \leq_1 \mathfrak{t}(\overline{a})[s]$. It follows that for large enough s, $\mathfrak{t}(\overline{a})[s] = \mathfrak{t}(\overline{a})$, and that $\{(\mathcal{C}[s], \mathfrak{t}[s]) : s \in \omega\}$ is a 1-approximation to $\mathcal{C}$. □

5. The low property

Recall from 1.3 the definition of the low property.

The following lemma will be necessary in the proof that ∞-diagonalizability implies that the low property fails. A similar lemma was proved by Harris and the author in [HM] for the class of Boolean algebras.

Lemma 5.1. *If $\mathbb{K}$ is axiomatizable by a Π^c_2 sentence and has a computable 1-back-and-forth structure, then $0'$ can effectively list all the structures in*

$$\{\mathcal{A}_{(1)} : \mathcal{A} \in \mathbb{K}, \mathcal{A} \text{ computable}\}.$$

Proof. The main difficulty to get an effective list of $\{\mathcal{A} : \mathcal{A} \in \mathbb{K}, \mathcal{A} \text{ computable}\}$ is, of course, that we cannot list all total computable functions. With an oracle $0'$, however, if we have a non-total function, we will eventually find out. By the time we find out we would have built part of a structure, which $0'$ now needs to recycle into another structure within the class. Furthermore, $0'$ will also be able to get the structural jumps of these structures too. Here are the details.

Write the Π^c_2 axiom defining $\mathbb{K}$ as $\bigwedge_{i\in\omega} \forall \overline{y}_i \theta_i(\overline{y}_i)$ where each θ_i is a Σ^c_1 formula. First, we observe that if $\mathcal{A}$ is a computable structure in $\mathbb{K}$, then there is a computable approximation $\mathcal{A}[0] \subseteq \mathcal{A}[1] \subseteq \cdots$ to $\mathcal{A}$ which satisfies one instance of one conjunct θ_i at the time. That is, for every s, $\mathcal{A}[s+1] \models \theta_i(\overline{a})$, where $(i, \overline{a})$ is the sth pair consisting of some $i \in \omega$ and some $\overline{a}$ of length $|\overline{y}_i|$.

For each $e \in \omega$, using $0'$ we uniformly define a computable structure $\mathcal{B}^e_{(1)}$, such that, if $\{e\}$ is a computable approximation to $\mathcal{A} \in \mathbb{K}$ as above, then $\mathcal{B}^e \cong \mathcal{A}$.

Fix $e \in \omega$. For each s, let $\mathcal{A}[s]$ be the finite structure with index $\{e\}(s)$. If $\{e\}(s)$ is undefined, or not an index for the right kind of structure, we let $\mathcal{A}[s]$ be undefined. First, we can assume that $\{e\}$ is defined on an initial segment of ω, as, once one $\mathcal{A}[s]$ is undefined, we leave $\mathcal{A}[t]$ for $t > s$ undefined too. Second, we can assume that $\mathcal{A}[s] \subseteq \mathcal{A}[s+1]$, as we can pretend as if $\mathcal{A}[s+1]$ is undefined if not. Third, we can also assume that $\mathcal{A}[s+1]$ satisfies one new instance of one conjunct θ_i as explained above. If $\mathcal{A}[s]$ is defined for all s, we let $\mathcal{A} = \bigcup_s \mathcal{A}[s]$.

So far, we have that either $\mathcal{A}[s]$ is defined for all s, in which case we know $\mathcal{A}$ satisfies the Π^c_2 axiom defining $\mathbb{K}$, or $\mathcal{A}[s]$ is undefined from some point on. Notice we have not used $0'$ yet. We will now define a structure $\mathcal{B}^e_{(1)}$ (that we denote $\mathcal{B}_{(1)}$) uniformly in $0'$, such that if $\mathcal{A}[s]$ is defined for all s, then $\mathcal{B} = \mathcal{A}$, and if not, then $\mathcal{B}_{(1)}$ is some structure in $\mathbb{K}_{(1)}$. At stage s, we define an $\mathcal{L}_1 \restriction s$-structure $\mathcal{B}_{(1)}[s]$. At each stage, we also make sure that there is some $\tau_s \in \mathbf{bf}_1$ verifying the consistency of $\mathcal{B}_{(1)}[s]$. We say that $\tau \in \mathbf{bf}_1$ *verifies the consistency of* $\mathcal{B}_{(1)}[s]$ if

> $|\tau| = |\mathcal{B}[s]|$ and, if $\mathcal{B}[s]$ had Π_1-type τ, then for each $\overline{a} \in \mathcal{B}[s]^{<\omega}$, and all σ with Gödel index less than s, $\mathcal{B}_{(1)}[s] \models \varphi_\sigma(\overline{a})$ holds if and only if the restriction of τ to $\overline{a}$ is $\geq_1 \sigma$.

Notice that this is a computable property.

At stage s, suppose we have already defined $\mathcal{B}_{(1)}[s-1]$, that τ_{s-1} verifies the consistency of $\mathcal{B}_{(1)}[s-1]$, and that, so far, $\mathcal{B}[s-1] = \mathcal{A}[s-1]$. Ask $0'$ if $\mathcal{A}[s]$ is defined. If it is not we know we can define the rest of $\mathcal{B}_{(1)}$ in which ever way we want. We do the following:

> First, we declare that $\mathcal{B}[s-1]$ will have type τ_{s-1} within $\mathcal{B}$, which we know is consistent with the amount of $\mathcal{B}_{(1)}$ defined so far. We then define $\mathcal{B}_{(1)}[t]$ for all $t \geq s$, so that $\mathcal{B}_{(1)}$ is any $\mathcal{L}_1$ structure in $\mathbb{K}$ extending $\mathcal{B}[s-1]$ and giving $\mathcal{B}[s-1]$ Π_1-type τ_{s-1}. The fact that we can uniformly build a $\mathcal{L}_1$ structure in $\mathbb{K}$ with a tuple satisfying a prescribed Π_1-type was proved in [Mon10, Proposition 2.10] and in [Monb].

Suppose now that $\mathcal{A}[s]$ is defined. We will now attempt to define $\mathcal{B}[s] = \mathcal{A}[s]$ and use $0'$ to define $\mathcal{B}_{(1)}[s]$, unless we find out that $\mathcal{A}[t]$ is not defined for some $t \geq s$. For each $\sigma \in \mathbf{bf}_1$ with Gödel index less than s, we need to define $\boldsymbol{\varphi}_\sigma(\overline{a})$ for each $\overline{a} \in \mathcal{A}[s]$ of length $|\sigma|$. We start by defining an auxiliary predicate $R_\sigma(\overline{a})$ as follows. For each such σ and $\overline{a}$, let

$R_\sigma(\overline{a})$ hold if for all $t > s$ for which $\mathcal{A}[t]$ is defined, and all Π_1-formula $\psi(\overline{x}) \in \sigma$, we have $\mathcal{A}[t] \models \psi(\overline{a})$.

Notice that $0'$ can decide this Π^0_1 question. If $\mathcal{A}[t]$ is actually defined for all t, then this is exactly the definition of $\mathcal{A} \models \boldsymbol{\varphi}_\sigma(\overline{a})$, so we would want to define $\boldsymbol{\varphi}_\sigma(\overline{a})$ in $\mathcal{B}[s]$ this way. However, if $\mathcal{A}[t]$ is not defined for some t, then this $R_\sigma(\overline{a})$ gives us irrelevant information. Thus, before defining $\boldsymbol{\varphi}_\sigma(\overline{a})$ as $R_\sigma(\overline{a})$, we need to check that these answers are consistent. We ask $0'$ if there exists a $\tau \in \mathbf{bf}_1$ which verifies the consistency of $\mathcal{B}_{(1)}[s]$ if it were to be defined that way. If the answer is yes, we define $\boldsymbol{\varphi}_\sigma(\overline{a})$ according to $R_\sigma(\overline{a})$, and we let τ_s be such τ for future reference. If the answer is no, we know that there is some inconsistency and hence that some $\mathcal{A}[t]$ is undefined. In this case we define $\mathcal{B}[t]$ for all $t \geq s$ exactly as we did above in the case when $\mathcal{A}[s]$ was undefined. □

Theorem 5.2. *Let $\mathbb{K}$ be a class of structures axiomatizable by a Π^c_2 sentence and with a computable 1-back-and-forth structure, and let $\mathbb{K}_{(1)}$ be as in Definition 4.3. If $\mathbb{K}_{(1)}$ is ∞-diagonalizable, then $\mathbb{K}$ does not have the low property.*

Proof. In Lemma 5.1 we show that $0'$ can list

$$\{A_{(1)} : \mathcal{A} \in \mathbb{K}, \mathcal{A} \text{ computable}\}.$$

Since $\mathbb{K}_{(1)}$ is ∞-diagonalizable, playing against that list, D can build a $0'$-computable structure $\mathcal{D}_{(1)} \in \mathbb{K}_{(1)}$ different from all those in the list. That is, $\mathcal{D}$, (the restriction of $\mathcal{D}_{(1)}$ to $\mathcal{L}$) has no computable copy. In [Mon09, Theorem 3.1] (see also [Mona, Lemma 6.3]) we show that if $0'$ computes a copy of the structural jump $\mathcal{D}_{(1)}$ of a structure, then $\mathcal{D}$ has a low copy. So, $\mathcal{D}$ witnesses that $\mathbb{K}$ does not have the low property. □

We now move towards the proof that if $\mathbb{K}_{(1)}$ is ∞-copyable, $\mathbb{K}$ has the low property. We start by reviewing the notion of true stage.

Definition 5.3. Fix a computable enumeration of $0'$: $\{k_0, k_1, ...\}$. We say that $r \in \omega$ is a *true stage* if $\forall s > r(k_s > k_r)$. Throughout the rest of the paper we will use $r_0 < r_1 < \cdots$ to denote the sequence of true stages for the enumeration of $0'$. We say that r *looks true at* $s \geq r$ if for all s_0 with $r < s_0 \leq s$, $k_{s_0} > k_r$.

In the following lemma we approximate $0'$-computable 1-approximations.

Lemma 5.4. *Suppose that $\mathcal{D}$ has a $0'$-computable 1-approximation. Then, there is a computable sequence of finite labeled structures $\{(\mathcal{D}[s], \mathfrak{t}[s]) : s \in \omega\}$ such that*

(D1) *$\{(\mathcal{D}[r_i], \mathfrak{t}[r_i]) : i \in \omega\}$ is a 1-approximation to $\mathcal{D}$, and*
(D2) *if s_0 looks true at s_1, then $(\mathcal{D}[s_0], \mathfrak{t}[s_0]) \leq_1 (\mathcal{D}[s_1], \mathfrak{t}[s_1])$.*

Sketch of the proof. Let $\{\hat{\mathcal{B}}_t : t \in \omega\}$ be a $0'$-computable 1-approximation to $\mathcal{D}$, and let Φ be the computable functional such that $\Phi^{0'}(t)$ gives the index for $\hat{\mathcal{B}}_t$. By slowing down this approximation and repeating outcomes, we can assume that the use to compute $\Phi^{0'}(t)$ is at most t. Also notice that the t-th true stage r_t can compute at least t bits of $0'$.

At each stage s we do the following: Let $r^s_0 < r^s_1 < ... < r^s_i = s$ be the sequence of stages that look true at s and let $K_s = \{k_0, ..., k_s\} \restriction_{k_s+1} \in 2^{k_s+1}$ be our current approximation to $K = 0'$. Before defining $\hat{\mathcal{D}}[s] = \Phi^{K_s}_s(i)$ we need to verify that our sequence will satisfy (D2). If $\hat{\mathcal{D}}[r^s_{i-1}] \leq_1 \Phi^{K_s}_s(i)$, then we define $\hat{\mathcal{D}}[s] = \Phi^K_s(i)$, otherwise we know s is not a true stage and we let $\hat{\mathcal{D}}[s] = \hat{\mathcal{D}}[r^s_{i-1}]$.

The rest of the verification is standard. □

In the next lemma we show that $\mathbb{K}$ has the low property under the assumption that $\mathbb{K}_{(1)}$ is copyable, rather than ∞-copyable. We prove this lemma first because the proof is simpler in terms of notation, and then the proof of Theorem 5.6 is just an adaptation of this proof.

Lemma 5.5. *Suppose that $\mathbb{K}$ has a computable 1-back-and-forth structure. Then, if $\mathbb{K}_{(1)}$ is copyable, $\mathbb{K}$ has the low property.*

Proof. Let $\mathcal{D} \in \mathbb{K}$ be a low structure; we will show that $\mathcal{D}$ has a computable copy. By relativizing this argument, we get that $\mathbb{K}$ has the low property.

We consider the version of the game $\mathsf{G}(\mathbb{K}_{(1)})$ where the players play 1-approximations to $\mathcal{C}$ and $\mathcal{D}$ rather than approximations to $\mathcal{C}_{(1)}$ and $\mathcal{D}_{(1)}$. It follows from Lemma 4.8 that these two games are equivalent, and hence we know that C has a winning strategy.

Since $\mathcal{D}$ is low, $\mathcal{D}$ has a $0'$-computable 1-approximation. Let $\{\hat{\mathcal{D}}[s] : s \in \omega\}$ be as in the lemma above.

The idea of the proof is that at each stage s, we will set up a finite sequence of moves by D and see what C answers. We will only let D play $\hat{\mathcal{D}}[t]$ at those values t that look true at the current stage. We will also decide how many times we want D to pass in between moves. The tension is produced by the fact that we have to produce a computable copy of $\mathcal{C}$, and hence when we find out that our guess to $\mathcal{D}$ was wrong we cannot change that part of $\mathcal{C}$ we built already.

At each stage s we will build a finite sequence $\gamma_s \in \omega^{<\omega}$ which we will use to determine a finite sequence of moves by D and then we will let $\hat{\mathcal{C}}[\gamma_s]$ be C's answer to those moves. We will do it in such a way that

$$(\forall s) \;\; \hat{\mathcal{C}}[\gamma_s] \leq_0 \hat{\mathcal{C}}[\gamma_{s+1}],$$

(i.e. $\mathcal{C}[\gamma_s] \subseteq \mathcal{C}[\gamma_{s+1}]$). This will allow us to define a computable structure $\mathcal{C} = \bigcup \mathcal{C}[s]$ forgetting about the values of $\mathfrak{t}^{\mathcal{C}}$.

Given a non-empty increasing sequence $\gamma \in \omega^{<\omega}$, we let $\hat{\mathcal{C}}[\gamma]$ be defined as follows: Let $t = last(\gamma)$, and let r_i^γ the i-th stage that looks true at stage $\gamma(i)$ if there is such a stage, and let r_i^γ be $\gamma(i)$ if there is not (i.e., if there are less than i stages that look true at $\gamma(i)$). Notice that if $\gamma(i) \geq r_i$, then $r_i^\gamma = r_i$. (Recall that r_i is the i-th true stage in the enumeration of $0'$.) We define $\hat{\mathcal{C}}[\gamma]$ to be the step t outcome of C's strategy after D has played $\hat{\mathcal{D}}[r_i^\gamma]$ at stage $\gamma(i)$ for $i < |\gamma| - 1$, and has passed on the moves that are not in the image of γ (by 'pass' we mean play the same structure played at the previous stage).

Observe that if $\delta \subseteq \gamma$, then $\hat{\mathcal{C}}[\delta] \leq_1 \hat{\mathcal{C}}[\gamma]$ because the sequence of moves by D in the definition of $\hat{\mathcal{C}}[\delta]$ is an initial segment of the sequence of moves in the definition of $\hat{\mathcal{C}}[\gamma]$, so C's answers are part of a 1-approximation to some structure. More generally, and for the same reason, we get that

$$\text{if} \;\; \delta \subseteq \gamma \;\; \text{and} \;\; s < \gamma(|\delta|) \quad \Longrightarrow \quad \hat{\mathcal{C}}[\delta^\frown s] \leq_1 \hat{\mathcal{C}}[\gamma].$$

Given $\delta \in \omega^{<\omega}$ we let $\mathcal{C}[\delta*]$ be the union of the structures $\mathcal{C}[\gamma^\frown s]$ for $s \in \omega$. So, $\mathcal{C}[\delta*]$ is C's structure in the case when D has played $\hat{\mathcal{D}}[r_i^s]$ at stages $\gamma(i)$ for $i < |\gamma| - 1$, and has passed ever after. Since C wins, we know this is a structure in $\mathbb{K}$.

Before defining the sequence $\{\gamma_s : s \in \omega\}$ we still need a couple more definitions. We say that a stage s_0 is *stable for* δ if the 1-bf-type in $\mathcal{C}[\delta*]$ of any tuple from $\mathcal{C}[\delta]$ has settled by stage s_0, or equivalently, if

$$(\forall s \geq s_0) \;\; \hat{\mathcal{C}}[\delta^\frown s_0] \;\equiv_1^{\mathcal{C}[\delta]}\; \hat{\mathcal{C}}[\delta^\frown s].$$

In the definition above we allow $\delta = \emptyset$, where $\mathcal{C}[\emptyset]$ is the empty structure.

We say that $\gamma \in \omega^{<\omega}$ is *good* if for all $i < |\gamma|$,

- $r_i = r_i^\gamma$ and
- $\gamma(i)$ is stable for $\gamma\restriction_i$.

Notice that being good is a Π^0_1 property, and hence if a string is not good, we will eventually find out.

Observe that if s_0 is stable for δ, and $\gamma \supseteq \delta^\frown s_0$, then there is a stage s_δ such that $\mathcal{C}[\gamma]$ embeds in $\mathcal{C}[\delta^\frown s_\delta]$ fixing $\mathcal{C}[\delta]$. The reason is that

$$\hat{\mathcal{C}}[\gamma] \geq_1 \hat{\mathcal{C}}[\delta^\frown s_0] \equiv_1^{\mathcal{C}[\delta]} \mathcal{C}[\delta*],$$

and hence, it follows from Lemma 4.6 that $\mathcal{C}[\gamma]$ embeds in $\mathcal{C}[\delta*]$ fixing $\mathcal{C}[\delta]$.

Our best hope would be to define γ_s to be good at all stages. However, this will not be possible, and we can only try to get γ_s to be as good as possible.

First, we say that s_0 *looks stable for* δ *at* ℓ if $(\forall s) s_0 \leq s \leq \ell \implies \hat{\mathcal{C}}[\delta^\frown s_0] \equiv_1^{\mathcal{C}[\delta]} \hat{\mathcal{C}}[\delta^\frown s]$.

We say that γ *looks good at* ℓ, if, for all $i < |\gamma|$,

(L1) r_i^γ is the ith stage that looks true at ℓ,
(L2) $\gamma(i)$ looks stable for $\gamma\restriction_i$ at ℓ, and
(L3) for every $\delta \subsetneq \gamma$, there is an $s_\delta \leq \ell$ such that $\mathcal{C}[\gamma]$ embeds in $\mathcal{C}[\delta^\frown s_\delta]$ fixing $\mathcal{C}[\delta]$.

The third item above will be used to recover from our mistakes at the moment when we find out that some γ_s is not good. Notice that if a string γ is good, it looks good at any sufficiently large stage ℓ (namely larger that all the s_δ for $\delta \subsetneq \gamma$). Thus, for any string γ, good or not, there is a stage ℓ at which either γ looks good at ℓ, or γ is been proved to be not good (i.e., either (L1) or (L2) above do not hold).

We are now ready to define our construction. The idea is to define γ_s so that it looks good at some large enough stage ℓ_s. At stage $s = 0$, non-uniformly we define γ_0 to be any string of length one which is good, and let ℓ_0 be any number larger than the one in γ_0. At stage $s+1$, suppose we have already defined γ_s and ℓ_s, and we now want to define γ_{s+1} and ℓ_{s+1}. Search for a stage ℓ, a number r with $\ell_s \leq r < \ell$, and an $i \leq |\gamma_s|$ such that

$$\gamma_s\restriction_i{}^\frown r \text{ looks good at } \ell,$$

and one of the following happens:

(C1) $i = |\gamma_s|$, or
(C2) $r_i^{\gamma_s}$ does not look true at ℓ, or
(C3) $\gamma_s(i)$ does not look stable for $\gamma_s\restriction_i$ at ℓ.

Define $\gamma_{s+1} = \gamma_s\restriction_i{}^\frown r$ and $\ell_{s+1} = \ell$.

We claim that will eventually find such ℓ, r and i. Let $i < |\gamma_s|$ be the largest such that $\gamma_s\restriction i$ is good. Then, there exists an r such that $\gamma_s\restriction_i{}^\frown r$ is good, and there exists ℓ such that $\gamma_s{}^\frown r$ looks good at ℓ. If $i = |\gamma_s|$ these ℓ, r and i work for us. Otherwise, $\gamma_s\restriction_{i+1}$ is not good and there must exists an ℓ where we see this is the case and either (C2) or (C3) holds.

We now need to show that $\mathcal{C}[\gamma_s] \subseteq \mathcal{C}[\gamma_{s+1}]$. In the case when $\gamma_{s+1} \supset \gamma_s$, we have that $\hat{\mathcal{C}}[\gamma_s] \leq_1 \hat{\mathcal{C}}[\gamma_{s+1}]$. Suppose now that $i < |\gamma_s|$, and let $\delta = \gamma_s\restriction_i$. Since γ_s looks good at ℓ_s, for s_δ as in (L3) we have that $\mathcal{C}[\gamma_s]$ embeds in $\mathcal{C}[\delta^\frown s_\delta]$ preserving $\mathcal{C}[\delta]$. Since $r \geq \ell_s$, we have that $r > s_\delta$ and that $\mathcal{C}[\delta^\frown s_\delta] \subseteq \mathcal{C}[\delta^\frown r]$. Since $\gamma_{s+1} = \delta^\frown r$, we get that $\mathcal{C}[\gamma_s]$ embeds in $\mathcal{C}[\gamma_{s+1}]$ preserving $\mathcal{C}[\gamma_s\restriction_i]$.

Using this chain of embeddings we can now define $\mathcal{C} = \bigcup_s \mathcal{C}[\gamma_s]$.

For each i, let t_i be the first stage such that $\gamma_{t_i}\restriction_i$ is good. Using standard arguments one can show by induction on i that t_i exists, that $|\gamma_{t_i}| = i$, and that $(\forall s \geq t_i)\ \gamma_s \supseteq \gamma_{t_i}$. We also note that for all $s > t_i$, the embedding from $\mathcal{C}[\gamma_s]$ into $\mathcal{C}[\gamma_{s+1}]$ preserves $\mathcal{C}[\gamma_{t_i}]$.

Consider now the sequence $\{\hat{\mathcal{C}}[t_i] : i \in \omega\}$. This sequence is a subsequence of C's answer in a game where D plays $\hat{\mathcal{D}}[r_i]$ at stage $\gamma_{t_{i+1}}(i)$ and passes at the other stages. Since $\{\hat{\mathcal{D}}[r_i] : i \in \omega\}$

is a 1-approximation to $\mathcal{D}$, we have that $\{\hat{\mathcal{C}}[t_i] : i \in \omega\}$ is a 1-approximation to a structure isomorphic to $\mathcal{D}$. Thus $\mathcal{C} \cong \mathcal{D}$. □

Theorem 5.6. *Suppose that $\mathbb{K}$ has a computable 1-back-and-forth structure. Then, if $\mathbb{K}_{(1)}$ is ∞-copyable, $\mathbb{K}$ has the low property.*

Proof. The proof is very similar to the one of the lemma above, but now C is using infinitely many strategies, one of which is correct. The only extra care we need take is that all the strategies must produce computable structures simultaneously. This requires modifying the notion of good string a little bit.

We start with a sequence $\{\hat{\mathcal{D}}[s] : s \in \omega\}$ exactly as above. At each stage s we will build a finite sequence $\gamma_s \in \omega^{<\omega}$ modifying the definition above slightly. Then, pretty much as above, we will use γ_s to determine a finite sequence of moves by D and then we will let $\hat{\mathcal{C}}^j[\gamma_s]$ be the jth structure in C's answer to those moves. We will do it in a way that, on one hand, if s is a true stage, then D only uses the values of $\hat{\mathcal{D}}[r]$ at true stages r (which are real steps towards a 1-approximation of $\mathcal{D}$), and, on the other hand, for every j, there exists an s_j such that

$$(\forall s \geq s_j)\ \ \mathcal{C}[\gamma_s] \subseteq \mathcal{C}[\gamma_{s+1}].$$

This latter part will allow us to define computable structures $\mathcal{C}^j = \bigcup_{s \geq s_j} \mathcal{C}[s]$ forgetting about the values of $\mathfrak{t}^{\mathcal{C}}$.

Given an increasing sequence $\gamma \in \omega^{<\omega}$ and $j < |\gamma|$, we define $\hat{\mathcal{C}}^j[\gamma]$ to be the jth structure produced by the step t $(= last(\gamma))$ outcome of C's strategy after D has played $\hat{\mathcal{D}}[r_i^\gamma]$ at stage $\gamma(i)$ for $i < |\gamma| - 1$, and has passed on the moves that are not in the image of γ, exactly as above.

Given $\delta \in \omega^{<\omega}$ we let $\mathcal{C}^j[\delta*]$ be the union of the structures $\mathcal{C}^j[\gamma^\frown s]$ for $s \in \omega$, exactly as above.

Before defining the sequence $\{\gamma_s : s \in \omega\}$ we still need a couple more definitions. For $j \leq |\delta|$, we say that a stage s_0 is *stable for j and δ* if the 1-bf-type within $\mathcal{C}^j[\delta*]$ of any tuple from $\mathcal{C}^j[\delta]$ has settled by stage s_0. Accordingly, we say that s_0 *looks stable for j and δ at ℓ* if $(\forall s) s_0 \leq s \leq \ell \implies \hat{\mathcal{C}}^j[\delta^\frown s_0] \equiv_1^{\mathcal{C}^j[\delta]} \hat{\mathcal{C}}^j[\delta^\frown s]$. (Notice that $\mathcal{C}^j[\delta]$ is not defined when $j = |\delta|$, but for the definition of stable to make sense we let $\mathcal{C}^{|\delta|}[\delta]$ be the empty structure.)

We say that $\gamma \in \omega^{<\omega}$ is *good* if for all $i < |\gamma|$,

- $r_i = r_i^\gamma$ and
- $\gamma(i)$ is stable for j and $\gamma \restriction_i$ for all $j \leq i$.

We say that γ *looks good at ℓ*, if, for all $i < |\gamma|$,

(L1) r_i^γ is the ith stage that looks true at ℓ,
(L2) for all $j \leq i$, $\gamma(i)$ looks stable for j and $\gamma \restriction_i$ at ℓ, and
(L3) for every $\delta \subsetneq \gamma$ and every $j \leq |\delta|$, there is an $s_{\delta,j} \leq t$ such that $\mathcal{C}^j[\gamma]$ embeds in $\mathcal{C}^j[\delta^\frown s_{\delta,j}]$ fixing $\mathcal{C}^j[\delta]$.

The rest of the construction is exactly as above. At stage $s+1$, search for a stage ℓ, a number r with $\ell_s \leq r < \ell$, and an $i \leq |\gamma_s|$ such that

$$\gamma_s \restriction_i{}^\frown r \text{ looks good at } \ell,$$

and one of the following happens:

(C1) $i = |\gamma_s|$, or
(C2) $r_i^{\gamma_s}$ does not look true at ℓ, or
(C3) for some $j \leq i$, $\gamma_s(i)$ does not look stable for j and $\gamma_s \restriction_i$ at ℓ.

Define $\gamma_{s+1} = \gamma_s \restriction_i {}^\frown r$ and $\ell_{s+1} = \ell$.

We will eventually find such ℓ, r and i for the same reason as above, namely that good sequences do exists of all lengths. Again, let t_i be the first stage such that $\gamma_{t_i} \restriction_i$ is good. By the same arguments as in the previous theorem, for $j < t_i$ and for all $s \geq t_i$, $\mathcal{C}^j[\gamma_s]$ is defined and $\mathcal{C}^j[\gamma_s] \subseteq \mathcal{C}^j[\gamma_{s+1}]$. Using this chain of embeddings we can now define $\mathcal{C}^j = \bigcup_{s \geq t_i} \mathcal{C}^j[\gamma_s]$.

We also note that for all $s > t_i \geq t_{j+1}$, the embedding from $\mathcal{C}^j[\gamma_s]$ into $\mathcal{C}^j[\gamma_{s+1}]$ preserves $\mathcal{C}^j[\gamma_{t_i}]$.

Consider now the sequence $\{\hat{\mathcal{C}}^j[t_i] : i \in \omega, i > j\}$. This the jth structure of C's answer in a game where D plays $\hat{\mathcal{D}}[r_i]$ at stage $\gamma_{t_{i+1}}(i)$ and passes the other stages. Since $\{\hat{\mathcal{D}}[r_i] : i \in \omega\}$ is a 1-approximation to $\mathcal{D}$, we have that, for some j, $\{\hat{\mathcal{C}}^j[t_i] : i \in \omega\}$ is a 1-approximation to a structure isomorphic to $\mathcal{D}$. Thus $\mathcal{C}^j$ is a computable copy of $\mathcal{D}$. □

6. Adding guesses to the games

For each computable ordinal α, we define a new version, $\mathsf{G}^\alpha(\mathbb{K})$, of the game that is in between the game $\mathsf{G}(\mathbb{K})$ and the game $\mathsf{G}^\infty(\mathbb{K})$. (For concreteness, the reader may assume α is a natural number for the rest of this section.)

In the game $\mathsf{G}^\alpha(\mathbb{K})$, player C still gets to enumerate infinitely many structures $\mathcal{C}^i$ as in $\mathsf{G}^\infty(\mathbb{K})$, but he has to point at the one he claims is isomorphic to $\mathcal{D}$, and he is allowed to use α-Turing jumps to choose which structure to point at. More concretely, the game $\mathsf{G}^\alpha(\mathbb{K})$ is played exactly as the game $\mathsf{G}^\infty(\mathbb{K})$, but C has to start by picking an index $e \in \omega$ for the procedure he is going to use to pick his structure $\mathcal{C}^i$.

Player D		$\mathcal{D}[0]$	$\subseteq$	$\mathcal{D}[1]$	$\subseteq$	$\mathcal{D}[2]$	$\cdots$	$\mathcal{D} = \bigcup_s \mathcal{D}[s]$
	e							
Player C	$\mathcal{C}^0[0]$	$\subseteq$	$\mathcal{C}^0[1]$	$\subseteq$	$\mathcal{C}^0[2]$	$\subseteq$	$\cdots$	$\mathcal{C}^0 = \bigcup_s \mathcal{C}^0[s]$
			$\mathcal{C}^1[0]$	$\subseteq$	$\mathcal{C}^1[1]$	$\subseteq$	$\cdots$	$\mathcal{C}^1 = \bigcup_s \mathcal{C}^1[s]$
						$\ddots$	$\vdots$	$\vdots$

After the game is finished, we decide who wins as follows:

(1) If for some i, $\mathcal{C}^i \notin \mathbb{K}$, then D wins.
(2) If for all i, $\mathcal{C}^i \in \mathbb{K}$, but $\mathcal{D} \notin \mathbb{K}$, then C wins.
(3) If $\mathcal{D} \in \mathbb{K}$, but $\{e\}^{X^{(\alpha)}}(0)\uparrow$, then D wins, where $X = \langle \mathcal{D}[s] : s \in \omega \rangle$. Otherwise, let $i_0 = \{e\}^{X^{(\alpha)}}(0)$.
(4) If $\mathcal{D}, \mathcal{C}^0, \mathcal{C}^1, \ldots \in \mathbb{K}$ and $\mathcal{D} \cong \mathcal{C}^{i_0}$, then C wins.
(5) If $\mathcal{D}, \mathcal{C}^0, \mathcal{C}^1, \ldots \in \mathbb{K}$ and $\mathcal{D} \ncong \mathcal{C}^{i_0}$, then D wins.

Definition 6.1. We say that $\mathbb{K}$ is *α-copyable* if C has a computable strategy in the game $\mathsf{G}^\alpha(\mathbb{K})$. We say that $\mathbb{K}$ is *α-diagonalizable* if D has a computable strategy in the game $\mathsf{G}^\alpha(\mathbb{K})$.

Observation 6.2. We note that for a class $\mathbb{K}$

$$\text{copyable} \Rightarrow \text{1-copyable} \Rightarrow \text{2-copyable} \Rightarrow \text{3-copyable} \Rightarrow \cdots \infty\text{-copyable},$$

and that

$$\text{diagonalizable} \Leftarrow \text{1-diagonalizable} \Leftarrow \text{2-diagonalizable} \Leftarrow \cdots \infty\text{-diagonalizable}.$$

For an ∞-copyable class $\mathbb{K}$, we define the *game ordinal* of a class $\mathbb{K}$ to be the least ordinal α such that $\mathbb{K}$ is α-copyable (relative to some X). We expect these games to give much finer information about the structure $\mathbb{K}$ than the games $\mathsf{G}(\mathbb{K})$ and $\mathsf{G}^\infty(\mathbb{K})$.

7. Linear orderings

In this section we show that the class of infinite linear orderings is 4-copyable (and hence ∞-copyable) and 2-diagnonalizable.

Lemma 7.1. *The class of infinite linear orderings with no maximal elements is copyable.*

Proof. Let us describe C's strategy. At each step s, C will keep track of an isomorphism $p\colon \mathcal{D}[s] \to \mathcal{C}[s]$ so that the image of p is always an initial segment of $\mathcal{C}[s]$. While D passes, C adds elements to the right end of $\mathcal{C}[s]$. If D adds an element to its structure which is to the right of all the previous elements, then C maps it through p to the least element in $\mathcal{C}[s]$ outside the image of p, if such an element exists, and adds one element if it does not exist. If D adds an element to $\mathcal{D}[s]$ which is not to the right of the rest, then C adds an element to $\mathcal{C}[s]$ in the appropriate place so that it can extend p to $\mathcal{D}[s]$.

Since $\mathcal{D}$ has no maximal elements, then all elements in $\mathcal{C}$ will eventually be part of the image of p, and we will get an isomorphism between $\mathcal{D}$ and $\mathcal{C}$. □

Lemma 7.2. *The class of linear orderings is 4-copyable.*

Proof. Player C will build structures

- $\mathcal{C}^0$,
- $\mathcal{C}^1$,
- $\mathcal{C}^{a,L}$ for each a in the domain of $\mathcal{D}$ (that is, in ω),
- $\mathcal{C}^{a,R}$ for each a in the domain of $\mathcal{D}$,
- $\mathcal{C}^{a,b}$ for each a, b in the domain of $\mathcal{D}$, and
- $\mathcal{C}^2$.

Each structure will be built using a different strategy under a different guess about $\mathcal{D}$. Whenever this guess is correct, we will get an isomorphic copy of $\mathcal{D}$, and it is going to take us 4 Turing jumps to guess corectly.

We build $\mathcal{C}^0$ exactly as in the previous lemma, under the assumption that $\mathcal{D}$ has no greatest element. We build $\mathcal{C}^1$ in a symmetric way, assuming $\mathcal{D}$ has no least element.

For each a in the domain of $\mathcal{D}$, we build $\mathcal{C}^{a,L}$ under the assumption that a is a limit point of $\mathcal{D}$ from the left, (that is, that $\forall x < a \exists y\ (x < y < a)$). All we do is build $\mathcal{C}^{a,L}$ as a sum of two linear orderings, the right one copying $\mathcal{D}_{(\geq a)}$ step by step (and passing when D passes), and the left one copying $\mathcal{D}_{(<a)}$ using the strategy of the previous lemma (where C never passes), since we are under the assumption that $\mathcal{D}_{(<a)}$ has no maximal element. We build $\mathcal{C}^{a,R}$ analogously under the assumption that a is a limit point from the right. These are Π^c_2 guesses.

For each $a, b \in \mathcal{D}$, we build $\mathcal{C}^{a,b}$ under the assumption that the interval $(a,b)_{\mathcal{D}}$ is isomorphic to $\omega + \omega^*$. We build $\mathcal{C}^{a,b}$ as a sum of three linear orderings, the left one copying $\mathcal{D}_{(\leq a)}$ step by step, the right one copying $\mathcal{D}_{(\geq b)}$ step by step, and the middle being a copy of $\omega + \omega^*$ adding one new element at each step. These are Π^c_3 guesses.

Suppose that none of these guesses is right. So, $\mathcal{D}$ has a least element, a greatest element and any other element has a predecessor and a successor. Thus $\mathcal{D}$ has the form $\omega + \zeta \cdot \mathcal{L} + \omega^*$ for some linear ordering $\mathcal{L}$ (where ζ is the ordering of the integers). Furthermore, since $\mathcal{D}$ has no segments isomorphic to $\omega + \omega^*$, $\mathcal{L}$ cannot have two adjacent elements, nor it can have a least or a greatest element. It follows that the only possibility left is that $\mathcal{D} \cong \omega + \zeta \cdot \eta + \omega^*$ (where η is the ordering of the rationals). Just build $\mathcal{C}^2$ to be $\omega + \zeta \cdot \eta + \omega^*$.

Guessing that all the previous guesses are wrong is Π^c_4, so within 4 Turing jumps of the diagram of $\mathcal{D}$ we can choose the correct strategy to copy $\mathcal{D}$. □

Note the similarity in flavor of the above argument and Rosenstein's theorem that every computable linear ordering has a computable sub-ordering isomorphic to either ω, ω^*, $\omega + \omega^*$ or $\omega + \zeta \cdot \eta + \omega^*$.

Lemma 7.3 (Kach–Montalbán, with ideas from Jockusch and Soare [JS91])**.** *The class of linear orderings is diagonalizable.*

Proof. The strategy for D is the following:

We pledge that $\mathcal{D}$ will be isomorphic to either ω or $m + \omega^*$ for some $m \in \omega$. Wait for one element c to appear in $\mathcal{C}$, which we fix for the rest of the construction.

At each stage s, let $n[s]$ be the number of predecessors of c in $\mathcal{C}[s]$.

For our construction, we will use a restraint-function $m[s]$; we will never add elements to $\mathcal{D}[s]$ below its $m[s]$'th element. Start by setting $m[s] = 0$. At all stages we make sure $m[s] \leq n[s]$. At stage s:

(1) If C enumerates an element to the right of c at stage s, we enumerate one more element to the right of the $m[s]$'th element of $\mathcal{D}[s]$. That is, we take one step towards enumerating $m[s] + \omega^*$. We note that if this happens infinitely often without changing the value of $m[s]$, then c in $\mathcal{C}$ will have infinitely many elements to its right and $n[s]$ many to its left, while no element in $m[s] + \omega^*$ has this property.

(2) If C enumerates an element to the left of c, we pass, unless $n[s]$ becomes greater than $|\mathcal{D}[s]|$. In this case we take this opportunity to take one step towards building $\mathcal{D} = \omega$: We re-set $m[s] = |\mathcal{D}_s|$, ensuring that every element enumerated in the future is to the right of all elements of $\mathcal{D}[s]$. Notice that if this happens infinitely often, we will get that $\mathcal{D} = \omega$ and that c has infinitely many elements to its left in $\mathcal{C}$.

If C is actually building an infinite linear ordering, then either (2) will occur infinitely often, or from some point on (1) will occur infinitely often without $m[s]$ changing. We already argued that in either case we are getting $\mathcal{D} \ncong \mathcal{C}$. □

Lemma 7.4. *The class of linear orderings is 2-diagonalizable.*

Sketch of the proof. It is well know that a $0''$ guess can be viewed as "the least one that repeats infinitely often." More formally, given $e \in \omega$ and $X \in 2^\omega$, one can build computable functions z and g such that

- there is at most one ℓ such that $z(s) = \ell$ for infinitely many s,
- such an ℓ exists if and only if $\{e\}^{X''}(0) \downarrow$, and
- if $\ell_0 = z(s)$ for infinitely many s, then $\{e\}^{X''}(0) = g(\ell_0)$.

A proof of this fact is given in [HM]. The proof is actually uniform in e and X.

To define D's strategy, we start by waiting to see what number e C plays, and then considering the functions z and g as above. The strategy will have infinitely many requirements R_ℓ working under the assumption that $z(s) = \ell$ for infinitely many s, and trying to diagonalize against $\mathcal{C}^{g(\ell)}$. For $\ell_0 < \ell_1$, R_{ℓ_0} will have higher priority than R_{ℓ_1}. At each stage s, we act for requirement $R_{z(s)}$, and initialize R_ℓ for all $\ell > z(s)$.

Each requirement R_ℓ will choose an element $c_\ell \in \mathcal{C}$, and use auxiliary values $n_\ell[s]$ and $m_\ell[s]$. The one new condition we will require is that if $\ell_0 < \ell_1$, then $m_{\ell_0}[s] < m_{\ell_1}[s]$ (if both are defined). The first time R_ℓ acts (after its last initialization) it wants to choose $m_\ell[s] > \max\{m_j[s] : j < \ell\}$. So, R_ℓ won't start acting until $\mathcal{C}[s]$ has more than $\max\{m_j[s] : j < \ell\}$ many elements, and only then it will chose c_ℓ to be the last element in $\mathcal{C}[s]$, $n_\ell[s]$ to be the number of elements in $\mathcal{C}[s]$ to the left of c_ℓ and $m_\ell[s] = n_\ell[s]$. At the rest of the stages for which $z(s) = \ell$, R_ℓ will act exactly as the strategy for D in the lemma above.

Suppose that $\{e\}^{X''}(0) \downarrow$ and $z(s) = \ell_0$ for infinitely many s. Standard arguments show that there will be a stage after which R_{ℓ_0} is never initialized again, and the requirements to

the right of ℓ_0 do not injure R_{ℓ_0}. Arguing like in the lemma above we will get that either $\lim_s m_{\ell_0}[s] \downarrow = m$, $\mathcal{D} = m + \zeta$ and c_{ℓ_0} has at least m elements to its left and infinitely many to its right, or $\lim_s m_{\ell_0}[s] = \infty$, $\mathcal{D} = \omega$ and c_{ℓ_0} has infinitely many elements to its left. □

References

[DJ94] Rod Downey and Carl G. Jockusch. Every low Boolean algebra is isomorphic to a recursive one. *Proc. Amer. Math. Soc.*, 122(3):871–880, 1994.

[GN02] S. S. Goncharov and Dzh. Naĭt. Computable structure and antistructure theorems. *Algebra Logika*, 41(6):639–681, 757, 2002.

[HM] Kenneth Harris and Antonio Montalbán. Boolean algebra approximations. Submitted for publication.

[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.

[JS91] Carl G. Jockusch, Jr. and Robert I. Soare. Degrees of orderings not isomorphic to recursive linear orderings. *Ann. Pure Appl. Logic*, 52(1-2):39–64, 1991. International Symposium on Mathematical Logic and its Applications (Nagoya, 1988).

[KM11] Asher M. Kach and Antonio Montalbán. Cuts of linear orders. *Order*, 28(3):593–600, 2011.

[KS00] Julia F. Knight and Michael Stob. Computable Boolean algebras. *J. Symbolic Logic*, 65(4):1605–1623, 2000.

[Mil01] Russell Miller. The Δ^0_2-spectrum of a linear order. *J. Symbolic Logic*, 66(2):470–486, 2001.

[Mona] Antonio Montalbán. Rice sequences of relations. To appear in the Philosophical Transactions A.

[Monb] Antonio Montalbán. When hyperarithmetic is recursive. In Preparation.

[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.

[Nur74] A. T. Nurtazin. *Computable classes and algebraic criteria for autostability*. PhD thesis, Institute of Mathematics and Mechanics, Alma-Ata, 1974.

Department of Mathematics, University of California, Berkeley, USA

E-mail address: antonio@math.berkeley.edu

URL: www.math.berkeley.edu/∼antonio