ON COMPUTABLE CATEGORICITY

RODNEY G. DOWNEY, ASHER M. KACH, STEFFEN LEMPP,
AND DANIEL D. TURETSKY

Abstract. We study the notion of computable categoricity of computable structures, comparing it especially to the notion of relative computable categoricity and its relativizations.

In particular, we show that every 1-decidable computably categorical structure is relatively Δ_2^0-categorical, but that there is a computably categorical structure that is not even relatively arithmetically categorical.

We also study the complexity of various index sets associated with computable categoricity and relative computable categoricity, though the index set complexity of the computably categorical structures remains open. Finally, we introduce and study a variation of relative computable categoricity, comparing it to both computable categoricity and relative computable categoricity and its relativizations.

1. Introduction

This paper contributes to computable (effective) model theory, a subject devoted to understanding structures with effective presentations. We recall that a structure is *computable* if it has a presentation where the universe and atomic diagram are computable. A very long-term program in computable model theory is to align syntactic complexity of (aspects of) computable structures with computability-theoretic properties. As an illustration of this program, we recall the notion of computable categoricity.

Definition 1.1. A computable structure $\mathcal{S}$ is *computably categorical* if between any two computable presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{S}$, there is a computable isomorphism.[1]

As is well-known, the countable dense linear order without endpoints is computably categorical using Cantor's back-and-forth argument. The model-theoretic view is to try to put this computable categoricity result in some larger framework. The idea is that perhaps there is some deeper explanation of this categoricity result. As it turns out, there is indeed such a deeper reason for countable dense linear

2010 *Mathematics Subject Classification.* 03C57.

Key words and phrases. computable categoricity, relative computable categoricity.

Downey's and Kach's research was supported by a grant from the Marsden Fund of New Zealand. Lempp's research was partially supported by NSF grant DMS-0555381 and Grant # 13407 by the John Templeton Foundation entitled "Exploring the Infinite by Finitary Means". Turetsky's research was supported by the Graduate School of the University of Wisconsin-Madison through a research assistantship. Lempp and Turetsky would like to thank Victoria University of Wellington for its hospitality during the time this work was carried out. Finally, all the authors would like to thank Alexander Melnikov and Adam Day for interesting discussions, especially concerning Theorem 5.2 and Proposition 5.4.

[1]Note that unlike the notion of κ-categoricity in classical model theory, which is a property of *theories*, computable categoricity is a property of (computable) *structures*.

orders: These structures have a certain kind of computable Scott formula (family), making them relatively computably categorical, a strengthening of computable categoricity as follows.

Definition 1.2. A computable structure $\mathcal{S}$ is *relatively computably categorical* if between any two (possibly noncomputable) presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{S}$, there is an isomorphism computable in $\deg(\mathcal{A}) \cup \deg(\mathcal{B})$, where we identify a presentation with its atomic diagram; or, equivalently, if for any computable presentation $\mathcal{A}$ of $\mathcal{S}$ and any (possibly noncomputable) presentation $\mathcal{B}$ of $\mathcal{S}$, there is an isomorphism computable in $\deg(\mathcal{B})$.

We now state the following classical result of Goncharov. It can be viewed as a computable analog of the Scott Isomorphism Theorem.

Theorem 1.3 (Goncharov [10]). *The following are equivalent for a computable structure $\mathcal{S}$:*

(1) The structure $\mathcal{S}$ is relatively computably categorical.

(2) The structure $\mathcal{S}$ has a c.e. Scott family *of existential formulas over some fixed $\overline{c} \in S$, i.e., a c.e. family Φ of existential formulas over some fixed $\overline{c} \in S$ such that each $\overline{a} \in S$ satisfies some $\varphi \in \Phi$, and if $\overline{a}, \overline{b} \in S$ both satisfy the same $\varphi \in \Phi$ then they are automorphic.*

In other words, the orbits of $\mathcal{S}$ are effectively isolated *by existential formulas.*

(3) The structure $\mathcal{S}$ has a c.e. family Φ of existential formulas over some fixed $\overline{c} \in S$ such that each $\overline{a} \in S$ satisfies some $\varphi \in \Phi$, and if $\overline{a}, \overline{b} \in S$ both satisfy the same $\varphi \in \Phi$ then they satisfy the same existential formulas.

In other words, the existential types of $\mathcal{S}$ are effectively isolated *by existential formulas.*

The program of aligning computational properties of structures with effective syntactic properties goes back to the pioneering work of Goncharov (see [10]), Ash and Nerode (see [4]) and others, and is the theme of a monograph by Ash and Knight (see [2]).

Our paper sits squarely within this program. This paper is devoted to trying to understand computable categoricity and the extent to which computable categoricity aligns itself with effective infinitary Scott formulas via theorems like Theorem 1.3.

Another goal of this paper is to relate computable categoricity to definability in arithmetic. The fundamental results of Emil Post showed that computational complexity (as measured by the jump operator) goes hand in hand with syntactic definability (as measured by quantifier depth and arithmetical complexity). Post's Theorem says that the Σ^0_n-sets are the sets many-one reducible to $\emptyset^{(n)}$, and the Δ^0_{n+1}-sets are exactly the $\emptyset^{(n)}$-computable sets. Thus we would anticipate that there should be some alignment of arithmetic complexity with the syntactic definability (in arithmetic) of computable structures. A natural way to measure this is via index sets. In this vein, a rather long-standing open problem in computable model theory concerns the index set complexity of the computably categorical structures.

Question 1.4. What is the computational complexity of a computable structure being computably categorical (in terms of Kleene's arithmetical or analytical hierarchy)?

In this paper, we will establish the computational complexity of a computable structure being relatively computably categorical. We also establish the computational complexity of a computable structure being isomorphic to a fixed computably categorical or relatively computably categorical structure. Question 1.4 remains quite opaque: An easy upper bound is Π^1_1, since the definition of computable categoricity involves the existence of a classical isomorphism in the hypothesis. The best-known lower bound was shown by Walker White.

Theorem 1.5 (White [15])**.** *The index set of the computably categorical structures is Π^0_4-hard.*

Our contributions to this question are not deep; but, hopefully, our methods will ultimately lead to a solution to Question 1.4.

1.1. **A First Summary.** We start with our motivating questions:

- How does computable categoricity align itself with descriptive complexity of the structure (in the language of the structure)?
- How does computable categoricity align itself with computational complexity as measured by, for example, the index sets associated with the structures; that is, to definability in arithmetic?
- How are the considerations above affected by *stronger* effectivity considerations about the structure, i.e., beyond simple computable presentability? What happens if the structure is decidable or n-decidable for some n? Does this make any difference?

Before we give the formal definitions needed for our results, we offer an informal description of our findings. If we require 2-decidability, then computable categoricity aligns itself with a c.e. Scott family of existential formulas in the sense of Theorem 1.3 by another result of Goncharov (see Theorem 1.11). If we require 1-decidability, then computable categoricity aligns itself with a c.e. Scott family of Σ^c_2-formulas. The surprise is that computable categoricity with no extra decidability does not align itself with the existence of a c.e. Scott family of formulas in any finite level of the arithmetic hierarchy. This is the most difficult result of the paper and uses technology not used before in computable model theory or effective algebra.

1.2. **Definitions and Our Results in More Detail.** To state and demonstrate our results, we will need some further definitions. Ash extended Theorem 1.3 from relative computable categoricity to relative Δ^0_α-categoricity.

Definition 1.6. A computable structure $\mathcal{S}$ is

1. *relatively Δ^0_α-categorical* if between any two (possibly noncomputable) presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{S}$, there is an isomorphism which is $\Delta^0_\alpha(\mathcal{A} \oplus \mathcal{B})$; or, equivalently, if for any computable presentation $\mathcal{A}$ of $\mathcal{S}$ and any (possibly noncomputable) presentation $\mathcal{B}$ of $\mathcal{S}$, there is an isomorphism computable in $\Delta^0_\alpha(\mathcal{B})$;
2. *relatively (hyper)arithmetically categorical* if between any two (possibly noncomputable) presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{S}$, there is an isomorphism which is (hyper)arithmetical in $\deg(\mathcal{A}) \cup \deg(\mathcal{B})$; or, equivalently, if for any computable presentation $\mathcal{A}$ of $\mathcal{S}$ and any (possibly noncomputable) presentation $\mathcal{B}$ of $\mathcal{S}$, there is an isomorphism (hyper)arithmetical in $\mathcal{B}$.

Note the following observation:

Proposition 1.7. *A computable structure is relatively arithmetically categorical if and only if it is relatively Δ^0_n-categorical for some $n < \omega$.*

A computable structure is relatively hyperarithmetically categorical if and only if it is relatively Δ^0_α-categorical for some $\alpha < \omega_1^{CK}$.

We will sketch the proof of this proposition at the beginning of Section 4; it follows from the proof of Theorem 1.9 below.

Ash then relativized Goncharov's theorem using "computable Σ_α-formulas" (denoted as Σ^c_α-formulas):

Definition 1.8 (Ash [1]). We define by recursion on computable ordinals α the collections of Σ^c_α- and Π^c_α-formulas (in a computable language $\mathcal{L}$). Each such formula has only a finite number of free variables, though it may have infinitely many bound variables.

(1) A Σ^c_0- or Π^c_0-*formula* is a quantifier-free first-order $\mathcal{L}$-formula.
(2) A Σ^c_α-*formula*, for computable $\alpha > 0$, is (logically equivalent to) an infinite c.e. disjunction of formulas of the form $\exists\overline{x}\,\varphi(\overline{x},\overline{y})$ where each φ is a Π^c_β-formula for some $\beta < \alpha$ and $\overline{y}$ is the tuple of free variables.
(3) A Π^c_α-*formula*, for computable $\alpha > 0$, is (logically equivalent to) the negation of a Σ^c_α-formula.

Theorem 1.9 (Ash [1]). *The following are equivalent for a computable structure $\mathcal{S}$:*

(1) The structure $\mathcal{S}$ is relatively Δ^0_α-categorical.
*(2) The structure $\mathcal{S}$ has a Σ^0_α-*Scott family* of Σ^c_α-formulas over some fixed $\overline{c} \in S$, i.e., a Σ^0_α-family Φ of Σ^c_α-formulas over some fixed $\overline{c} \in S$ such that each $\overline{a} \in S$ satisfies some $\varphi \in \Phi$, and if $\overline{a}, \overline{b} \in S$ both satisfy the same $\varphi \in \Phi$ then they are automorphic.*

    In other words, the orbits of $\mathcal{S}$ are effectively isolated *by Σ^c_α-formulas.*
(3) The structure $\mathcal{S}$ has a c.e. family Φ of Σ^c_α-formulas over some fixed $\overline{c} \in S$ such that each $\overline{a} \in S$ satisfies some $\varphi \in \Phi$, and if $\overline{a}, \overline{b} \in S$ both satisfy the same $\varphi \in \Phi$ then they satisfy the same Σ^c_α-formulas.

    In other words, the Σ^c_α-types of $\mathcal{S}$ are effectively isolated *by Σ^c_α-formulas.*

One might at first guess that the notions of computable categoricity and relative computable categoricity coincide (although Theorem 1.5 and Theorem 1.3 already indicate that they cannot). However, if we add more computability-theoretic assumptions, then the two notions do coincide. These assumptions are specified in the following definitions.

Definition 1.10. A computable presentation $\mathcal{A}$ of a computable structure $\mathcal{S}$ is:

(1) *decidable* if the elementary diagram of $\mathcal{A}$ is computable;
(2) *n-decidable* if the Σ_n-elementary diagram of $\mathcal{A}$ (in the first-order language of $\mathcal{S}$) is computable.

If $\mathcal{S}$ is computably categorical, it is easy to see that some computable presentation of $\mathcal{S}$ is decidable (n-decidable) if and only if every computable presentation of $\mathcal{S}$ is decidable (n-decidable).

Goncharov showed that for 2-decidable structures, computable categoricity and relative computable categoricity coincide.

Theorem 1.11 (Goncharov [10])**.** *A 2-decidable structure is computably categorical if and only if it is relatively computably categorical.*

The assumption of 2-decidability cannot be dropped completely, as observed by Goncharov (see Theorem 4 of [11]); in fact, Kudinov showed that even 1-decidability is not sufficient to ensure computable categoricity and relative computable categoricity coincide.

Theorem 1.12 (Kudinov [12])**.** *There is a 1-decidable structure that is computably categorical but not relatively computably categorical.*

Our first main theorem shows that Goncharov's result "almost" holds for 1-decidable structures:

Theorem 1.13. *Any 1-decidable, computably categorical structure is relatively Δ^0_2-categorical.*

We will prove Theorem 1.13 in Section 2, and related results in Section 3.

The reader might perceive an emerging pattern here, namely, that weakening the decidability hypothesis by a quantifier level increases the level of relative categoricity by a jump. Thus, the natural guess would be that with 0-decidability (i.e., computable presentability), computable categoricity would imply relative Δ^0_3-categoricity. Alas, this attractive pattern is far from the truth. We show that without the additional assumption of 1-decidability, the notions of computable categoricity and relative computable categoricity are very different.

Theorem 1.14. *There is a computably categorical structure which is not relatively arithmetically categorical.*

We will prove Theorem 1.14 in Section 4. To do this, we construct such a structure with no Σ^0_n-Scott family for any finite n. We suspect that this theorem can be strengthened to answer the following question affirmatively:

Question 1.15. Is there a computably categorical structure that is not relatively hyperarithmetically categorical?

In contrast to the concept of computable categoricity, relative computable categoricity turns out to be relatively simple to classify in terms of its complexity: In Section 5, we prove the following.

Theorem 1.16. *The index set of the relatively computably categorical structures is Σ^0_3-complete.*

In Section 5, we also examine the index set complexity of certain fixed computably categorical and relatively computably categorical structures. In Section 6, we introduce the related notion of *relative computable categoricity above a degree* and examine its relationship with computable categoricity and relative computable categoricity and its relativizations. Whilst these results are not particularly difficult, they do shed more light on this material.

We refer the reader to [2] for background on computable model theory and effective algebra. Notation is more or less standard and generally follows [2] and [13].

2. Every 1-Decidable Computably Categorical Structure is Relatively Δ^0_2-Categorical

In this section, we show that computable categoricity implies relative Δ^0_2-categoricity amongst 1-decidable structures. The crux of the proof is contained within a lemma.

Definition 2.1. For a structure $\mathcal{A}$ and tuples $\overline{a}, \overline{p} \in A$, denote by $\Sigma_n\text{-tp}_{\overline{p}}(\overline{a})$ the set

$$\Sigma_n\text{-tp}_{\overline{p}}(\overline{a}) := \{\varphi(\overline{x}, \overline{y}) \in \Sigma_n : \mathcal{A} \models \varphi(\overline{a}, \overline{p})\}$$

and denote by $\Sigma^c_n\text{-tp}_{\overline{p}}(\overline{a})$ the set

$$\Sigma^c_n\text{-tp}_{\overline{p}}(\overline{a}) := \{\varphi(\overline{x}, \overline{y}) \in \Sigma^c_n : \mathcal{A} \models \varphi(\overline{a}, \overline{p})\},$$

where in both cases we consider only finitary (or infinitary, respectively) formulas in the language of the structure.

Lemma 2.2. *If $\mathcal{A}$ is computably categorical and 1-decidable, then there is a tuple $\overline{p} \in A$ such that distinct Σ_1-types over $\overline{p}$ are incomparable under inclusion, and for any $\overline{a}, \overline{a}' \in A$, if $\Sigma_1\text{-tp}_{\overline{p}}(\overline{a}) = \Sigma_1\text{-tp}_{\overline{p}}(\overline{a}')$, then $\Sigma^c_2\text{-tp}_{\overline{p}}(\overline{a}) = \Sigma^c_2\text{-tp}_{\overline{p}}(\overline{a}')$.*

Note here that for a tuple $\overline{a}$, the (finitary, first-order) Σ_1-type determines the (computable infinitary) Σ^c_1-type as well as the (classically infinitary) Σ^i_1-type. (This relationship, of course, fails at higher levels.)

Proof of Lemma 2.2. We build a computable presentation $\mathcal{B}$ isomorphic to $\mathcal{A}$ and attempt to make $\mathcal{B}$ not computably isomorphic to $\mathcal{A}$. The amount of $\mathcal{B}$ constructed when we consider the computable function φ_e witnessing $\mathcal{A}$ and $\mathcal{B}$ being computably isomorphic will determine the parameter $\overline{p}$.

In order to ensure $\mathcal{A}$ and $\mathcal{B}$ are classically isomorphic, we build an isomorphism $F : \mathcal{B} \to \mathcal{A}$ in a Δ^0_2-manner. We build $\mathcal{B}$ by constructing its atomic diagram in stages. At each stage s, we enumerate the next atomic sentence true about $\mathcal{A}$ into the atomic diagram of $\mathcal{B}$ as determined by the isomorphism F (as approximated at stage s).

Let $\{\psi_i\}_{i\in\omega}$ be a computable enumeration of all Σ_1-formulas in the language of $\mathcal{A}$.

Strategy Defeating φ_e: We fix a partial computable function $\varphi_e : B \to A$ and seek to ensure that φ_e is not an isomorphism.

Let s_0 be the stage at which this strategy is initialized. This strategy takes no action until a stage $s_1 > s_0$ when $B_{s_0} \subseteq \operatorname{dom} F_{s_1}$ and $A_{s_0} \subseteq \operatorname{range} F_{s_1}$. We then let $\overline{b}_0 := B_{s_1}$ and restrain the strategy, in the sense that $F \restriction \overline{b}_0$ cannot be changed by this strategy. At every stage s after becoming active, before we enumerate the next sentence into the atomic diagram of $\mathcal{B}$, we look for an opportunity to change F in such a way that it still extends to an isomorphism, but such that $F \circ \varphi_e^{-1}$ is guaranteed not to be an automorphism of $\mathcal{A}$ (ensuring that if F is an isomorphism, as it will be, then φ_e is not an isomorphism). We will find such opportunities if the types do not obey the conclusion of the lemma.

Before describing the strategy, we note the following. For any stage $t > s_1$, suppose $\overline{b}$ is a tuple from the domain of $\mathcal{B}_t$, and let $\delta_t(\overline{b}_0, \overline{b}, \overline{f})$ be the atomic diagram of $\mathcal{B}_t$, where $\overline{f} := B_t \backslash (\overline{b}_0 \cup \overline{b})$. Suppose $\overline{a} \in A$. At a stage $s > s_1$, we can redefine F to map $\overline{b}$ to $\overline{a}$ without changing $F \restriction \overline{b}_0$ if and only if $\mathcal{A} \models \exists \overline{x} \left[\delta_{s-1}(F_{s-1}(\overline{b}_0), \overline{a}, \overline{x})\right]$.

Here, we consider δ_{s-1} instead of δ_s because when this strategy acts at stage s, we have not yet enumerated the next sentence into the atomic diagram of B.

At a stage $s > s_1$, we consider every triple $(\overline{b}, \overline{b}', \overline{d})$ with $\overline{b}, \overline{b}' \in \operatorname{dom}(\varphi_{e,s})$ and $\overline{d} \in \operatorname{dom}(\varphi_{e,s}) \backslash (\overline{b}_0 \cup \overline{b})$. If this is the first stage at which we have considered this triple, we use 1-decidability to determine if there is a tuple $\overline{c} \in A^{|\overline{d}|}$ such that

$$\mathcal{A} \models \exists \overline{y} \left[\delta_{s-1}(F_{s-1}(\overline{b}_0), F_{s-1}(\overline{b}'), \overline{c}\,\overline{y})\right],$$

i.e., we ask whether we can redefine F by putting $F_s(\overline{b}) := F_{s-1}(\overline{b}')$ and $F_s(\overline{d}) := \overline{c}$ while respecting the restraint. If there is no such $\overline{c}$, we never consider this triple again (since we cannot redefine F, there is no point in considering it further). If there is such a $\overline{c}$, we search for one and assign it to this triple. When we consider this triple at future stages, this is the $\overline{c}$ to which we refer.

Then, for every triple $(\overline{b}, \overline{b}', \overline{d})$ being considered (along with its associated $\overline{c}$), we use 1-decidability to determine if

$$\mathcal{A} \models \left[\psi_i(F_{s-1}(\overline{b}), F_{s-1}(\overline{d})) \iff \neg\psi_i(F_{s-1}(\overline{b}'), \overline{c})\right] \tag{1}$$

for some $i < s$, i.e., we ask whether redefining F by putting $F_s(\overline{b}) := F_{s-1}(\overline{b}')$ and $F_s(\overline{d}) := \overline{c}$ might be useful. If so, fix some triple and some $i_0 < s$ for which (1) holds. We use 1-decidability to determine whether

$$\mathcal{A} \models \left[\psi_{i_0}(\varphi_e(\overline{b}), \varphi_e(\overline{d})) \iff \psi_{i_0}(F_{s-1}(\overline{b}), F_{s-1}(\overline{d}))\right], \tag{2}$$

i.e., we determine whether or not it is necessary to perform any action to prevent $F \circ \varphi_e^{-1}$ from being an automorphism. If (2) holds, we put $F_s(\overline{b}) := F_{s-1}(\overline{b}')$ and $F_s(\overline{d}) := \overline{c}$, and extend F_s such that $A_s \subseteq \operatorname{range} F_s$ and $B_s \subseteq \operatorname{dom} F_s$. If (2) fails, we put $F_s(\overline{b}) := F_{s-1}(\overline{b})$ and $F_s(\overline{d}) := F_{s-1}(\overline{d})$, and extend F_s such that $A_s \subseteq \operatorname{range} F_s$ and $B_s \subseteq \operatorname{dom} F_s$. Regardless of whether (2) holds or fails, we declare the strategy complete.

If (1) fails for all triples being considered and all $i < s$, we repeat the above process with $\exists \overline{y}\ \delta_s$ in place of ψ_i. That is, for every triple $(\overline{b}, \overline{b}', \overline{d})$ and associated $\overline{c}$ being considered, we use 1-decidability to determine if

$$\mathcal{A} \models \neg\exists \overline{y} \left[\delta_s(F_{s-1}(\overline{b}_0), F_{s-1}(\overline{b}'), \overline{c}\,\overline{y})\right], \tag{3}$$

i.e., we ask whether we will lose the ability to redefine F after we enumerate the next atomic sentence into the diagram of $\mathcal{B}$. If (3) fails for every triple, we will not lose the ability to redefine F, so we leave F alone and take no further action for this strategy at stage s.

If (3) holds for some triple, fix a triple for which it holds. We will lose the ability to redefine F, so we use 1-decidability to determine if

$$\mathcal{A} \models \exists \overline{y} \left[\delta_s(\varphi_e(\overline{b}_0), \varphi_e(\overline{b}), \varphi_e(\overline{d})\,\overline{y})\right], \tag{4}$$

i.e., we determine whether or not it is necessary to perform any action to prevent $F \circ \varphi_e^{-1}$ from being an automorphism. If (4) holds, we put $F_s(\overline{b}) := F_{s-1}(\overline{b}')$ and $F_s(\overline{d}) := \overline{c}$ and extend F_s such that $A_s \subseteq \operatorname{range} F_s$ and $B_s \subseteq \operatorname{dom} F_s$. If (4) fails, we put $F_s(\overline{b}) := F_{s-1}(\overline{b})$ and $F_s(\overline{d}) := F_{s-1}(\overline{d})$, and extend F_s such that $A_s \subseteq \operatorname{range} F_s$ and $B_s \subseteq \operatorname{dom} F_s$. Regardless of whether (4) holds or fails, we declare the strategy complete.

The strategy has two outcomes: `wait` and `stop`. Of course, these correspond to whether the strategy has been declared completed.

Construction: We put these strategies on a tree, performing a straightforward finite-injury argument in the usual manner. At each stage, the visited strategies on the tree act in priority order. After they have acted, if no strategy defined F_s, we define F_s by extending F_{s-1} to include A_s and B_s in the range and domain, respectively. Then the global strategy building $\mathcal{B}$ acts by taking the next atomic sentence $\theta_s(\overline{a})$ true about $\mathcal{A}$ and enumerating $\theta_s(F_s(\overline{a}))$ into the atomic diagram of $\mathcal{B}$.

Verification: We verify $F := \lim_s F_s$ exists and is an isomorphism. Consequently, there will be a (least) k such that $\varphi_k : \mathcal{B} \to \mathcal{A}$ is a computable isomorphism. Let σ be the strategy for φ_k along the true path, and let $\overline{b}_0$ be the restraint of σ. We show the desired relationships between the types of tuples of $\mathcal{A}$ using $\overline{p} := F(\overline{b}_0)$.

Claim 2.2.1. The function $F := \lim_s F_s$ exists and is an isomorphism.

Proof. The existence of F follows from the fact that, if a strategy redefines F on an element (in either the domain or the range), then no lower-priority strategy can redefine F on that element. Thus, by induction, the function F can change only finitely many times on any element (in either the domain or the range).

By construction, the function F is surjective and respects atomic sentences. Thus, it is injective (as equality is an atomic sentence) and so an isomorphism. □

Claim 2.2.2. If a strategy for defeating φ_e is along the true path and declared complete, then φ_e is not an isomorphism.

Proof. Let $(\overline{b}, \overline{b}', \overline{d})$ be the triple we act for. Note that $F(\overline{b}) = F_s(\overline{b})$ and $F(\overline{d}) = F_s(\overline{d})$.

If we act because of some ψ_{i_0}, then regardless of whether (2) holds, we have

$$\mathcal{A} \models \left[\psi_{i_0}(\varphi_e(\overline{b}), \varphi_e(\overline{d})) \iff \neg\psi_{i_0}(F_s(\overline{b}), F_s(\overline{d}))\right].$$

If we act because of δ_s, then regardless of whether (4) holds, we have

$$\mathcal{A} \models \exists \overline{y} \left[\delta_s(\varphi_e(\overline{b}_0), \varphi_e(\overline{b}), \varphi_e(\overline{d})\overline{y})\right] \iff \neg\exists \overline{y} \left[\delta_s(F_s(\overline{b}_0), F_s(\overline{b}), F_s(\overline{d})\overline{y})\right].$$

Thus, in either case, we have that $F \circ \varphi_e^{-1}$ is not an automorphism. □

Claim 2.2.3. The Σ_1-types over $\overline{p}$ are incomparable under inclusion.

Proof. Towards a contradiction, we suppose that there are $\overline{a}, \overline{a}' \in A$ with

$$\Sigma_1\text{-tp}_{\overline{p}}(\overline{a}) \subsetneq \Sigma_1\text{-tp}_{\overline{p}}(\overline{a}'). \tag{5}$$

Consider any stage $s+1$ at which σ is visited such that $F_s(\overline{b}) = \overline{a}$ and $F_s(\overline{b}') = \overline{a}'$ for some $\overline{b}, \overline{b}' \in \operatorname{dom}(\varphi_{k,s})$. Then at such a stage, it will always be possible for σ to define $F_{s+1}(\overline{b}) = \overline{a}'$.

Note that (5) is equivalent to $\Sigma_1\text{-tp}(\overline{p}\,\overline{a}) \subsetneq \Sigma_1\text{-tp}(\overline{p}\,\overline{a}')$. Since $F \circ \varphi_k^{-1}$ is an automorphism, we have

$$\Sigma_1\text{-tp}(\varphi_k(\overline{b}_0)\,\varphi_k(\overline{b})) \subsetneq \Sigma_1\text{-tp}(\overline{p}\,\overline{a}').$$

Fix a formula ψ_i true of $\overline{p}\,\overline{a}'$ but false of $\overline{p}\,\overline{a}$. Then at any stage $s > i$ when the strategy considers the triple $(\overline{b}_0\,\overline{b}, \overline{b}_0\,\overline{b}', \emptyset)$, it will redefine $F(\overline{b}) = \overline{a}'$ to defeat φ_k, contrary to our choice of k. □

Claim 2.2.4. For any tuples $\overline{a}, \overline{a}' \in A$, if $\Sigma_1\text{-tp}_{\overline{p}}(\overline{a}) = \Sigma_1\text{-tp}_{\overline{p}}(\overline{a}')$ then $\Sigma_2^c\text{-tp}_{\overline{p}}(\overline{a}) = \Sigma_2^c\text{-tp}_{\overline{p}}(\overline{a}')$.

Proof. Suppose $\Sigma_1\text{-tp}_{\overline{p}}(\overline{a}) = \Sigma_1\text{-tp}_{\overline{p}}(\overline{a}')$, or equivalently $\Sigma_1\text{-tp}(\overline{p}\,\overline{a}) = \Sigma_1\text{-tp}(\overline{p}\,\overline{a}')$. By symmetry, it suffices to show

$$\Sigma_2^c\text{-tp}(\overline{p}\,\overline{a}) \subseteq \Sigma_2^c\text{-tp}(\overline{p}\,\overline{a}').$$

Fix a formula $\exists \overline{x}\, \chi(\overline{p}\,\overline{a}, \overline{x}) \in \Sigma_2^c\text{-tp}(\overline{p}\,\overline{a})$ with $\chi \in \Pi_1^c$ and a witness $\overline{g} \in A$ so that $\mathcal{A} \models \chi(\overline{p}\,\overline{a}, \overline{g})$. We show $\exists \overline{x}\, \chi(\overline{p}\,\overline{a}, \overline{x}) \in \Sigma_2^c\text{-tp}(\overline{p}\,\overline{a}')$.

Consider a stage $s > s_1$ when σ is visited, $F(\overline{b}) = \overline{a}, F(\overline{b}') = \overline{a}'$ and $F(\overline{d}) = \overline{g}$ have converged, and $\overline{b}_0, \overline{b}, \overline{b}', \overline{d} \in \operatorname{dom}(\varphi_{k,s})$. Since

$$\mathcal{A} \models \exists \overline{x}\, \delta_s(\overline{p}, \overline{a}, \overline{g}\,\overline{x}),$$

from $\Sigma_1\text{-tp}(\overline{p}\,\overline{a}) = \Sigma_1\text{-tp}(\overline{p}\,\overline{a}')$, we have

$$\mathcal{A} \models \exists \overline{c}\, \exists \overline{y}\, \delta_s(\overline{p}, \overline{a}', \overline{c}\,\overline{y}).$$

Thus, there will be a $\overline{c}$ assigned to the triple $(\overline{b}_0\,\overline{b}, \overline{b}_0\,\overline{b}', \overline{d})$. Since σ is never declared complete (by Claim 2.2.2), there is never a stage $t > s$ when

$$\mathcal{A} \models \neg \exists \overline{y}\, [\delta_t(\overline{p}, \overline{a}', \overline{c}\,\overline{y})].$$

Thus σ will never lose the ability to define $F(\overline{b}) = \overline{a}'$ and $F(\overline{d}) = \overline{c}$.

If there were some ψ_i such that

$$\mathcal{A} \models \psi_i(\overline{p}\,\overline{a}', \overline{c}) \wedge \neg \psi_i(\overline{p}\,\overline{a}, \overline{g}),$$

then at some stage when we consider ψ_i, the strategy σ would be able to defeat φ_k, contrary to our choice of k.

Thus $\mathcal{A} \models \chi(\overline{p}\,\overline{a}', \overline{c})$. We conclude $\exists \overline{x} \chi(\overline{p}\,\overline{a}', \overline{x}) \in \Sigma_2^c\text{-tp}(\overline{p}\,\overline{a}')$ as desired. □

This completes the proof of Lemma 2.2. □

We are now ready to prove the main theorem of this section:

Theorem 1.13. *Any 1-decidable, computably categorical structure $\mathcal{A}$ is relatively Δ_2^0-categorical.*

Proof. Fix the parameters $\overline{p}$ from the above lemma.

For each $\overline{a} \in A$, let $\chi_{\overline{a}}(\overline{x})$ be the infinitary formula

$$\chi_{\overline{a}}(\overline{x}) := \bigwedge_{\substack{\psi \in \Pi_1(\overline{p}) \\ \mathcal{A} \models \psi(\overline{a})}} \psi(\overline{x}),$$

i.e., the conjunction of all first-order Π_1-formulas (with parameters from $\overline{p}$) true of $\overline{a}$. As a consequence of 1-decidability, this is a Π_1^c-formula.

We show that the family of formulas $\{\chi_{\overline{a}}(\overline{x})\}_{\overline{a} \in A}$ constitutes a Scott family. By Theorem 1.9, it suffices to show that they isolate the Σ_2^c-types. We therefore suppose $\mathcal{A} \models \chi_{\overline{a}}(\overline{a}')$ and show $\Sigma_2^c\text{-tp}_{\overline{p}}(\overline{a}') = \Sigma_2^c\text{-tp}_{\overline{p}}(\overline{a})$. If $\mathcal{A} \models \chi_{\overline{a}}(\overline{a}')$, then every Π_1-fact true of $\overline{a}$ is true of $\overline{a}'$. Hence every Σ_1-fact true of $\overline{a}'$ is true of $\overline{a}$, i.e.,

$$\Sigma_1\text{-tp}_{\overline{p}}(\overline{a}') \subseteq \Sigma_1\text{-tp}_{\overline{p}}(\overline{a}).$$

By Lemma 2.2, it follows that $\Sigma_1\text{-tp}_{\overline{p}}(\overline{a}') = \Sigma_1\text{-tp}_{\overline{p}}(\overline{a})$. By Lemma 2.2 again, this implies that $\Sigma_2^c\text{-tp}_{\overline{p}}(\overline{a}') = \Sigma_2^c\text{-tp}_{\overline{p}}(\overline{a})$.

We conclude that the family of formulas $\{\chi_{\overline{a}}(\overline{x})\}_{\overline{a} \in A}$ constitutes a Scott family and so $\mathcal{A}$ is relatively Δ_2^0-categorical. □

3. Pushing on Isomorphisms and Results Related to Theorem 1.13

Theorem 1.13 raises questions about various ways in which the hypotheses can be weakened or the conclusion strengthened. In the next section, we investigate what happens when the 1-decidability hypothesis is weakened to just computability of the structure; in this section, we explore a number of other variations. None of the constructions are individually particularly difficult, so we only sketch their proofs. However, these constructions and many of the later constructions rely on the technique of *pushing on isomorphisms*. We illustrate this technique in isolation, demonstrating the existence of a computably categorical structure $\mathcal{S}$ that is not relatively computably categorical.

Theorem 3.1 (Goncharov [11, Theorem 4])**.** *There is a computable structure $\mathcal{A}$ that is computably categorical but not relatively computably categorical.*

Proof. Before discussing the formal details, we informally discuss the requisite ideas. The structure $\mathcal{A}$ will be a directed graph consisting of infinitely many finite connected components. Each component will consist of either two, three, or four cycles sharing only a single vertex v, termed the *root vertex*.

In order to prevent $\mathcal{A}$ from being relatively computably categorical, we diagonalize against all pairs $(\overline{c}, \Phi)$, where $\overline{c}$ is a finite tuple of elements from the universe of $\mathcal{A}$ and Φ is a c.e. family of existential formulas with parameters from $\overline{c}$. We create vertices v_1 and v_2 such that v_1 and v_2 are not automorphic, but Φ cannot distinguish them.

In order to ensure $\mathcal{A}$ is computably categorical, we construct a partial computable map f_j from $\mathcal{A}$ to $\mathcal{B}_j$ (the jth (partial) directed graph). If $\mathcal{A}$ and $\mathcal{B}_j$ are isomorphic, the map f_j will be an isomorphism.

More formally, we meet the following requirements to prevent relative computable categoricity:

$$\mathcal{R}_i : \text{The } i\text{th pair } (\overline{c}_i, \Phi_i) \text{ is not a Scott family for } \mathcal{A}.$$

We meet the following requirements to ensure computable categoricity:

$$\mathcal{S}_j : \text{If } \mathcal{A} \cong \mathcal{B}_j, \text{ then } f_j : \mathcal{A} \cong \mathcal{B} \text{ is a computable isomorphism.}$$

Strategy for Meeting $\mathcal{R}_i$ (In isolation): We take the following actions, being careful to use elements larger than those found in $\overline{c}_i$:

(1) Fix a *large* number ℓ and create two new root vertices v_1 and v_2.
(2) Attach a loop of length 2 and a loop of length 3ℓ to both v_1 and v_2 and a loop of length $3\ell + 1$ to v_1.
(3) For every formula $\varphi(x, \overline{c}_i) := (\exists \overline{y})[\psi(x, \overline{y}, \overline{c}_i)]$ in Φ_i, search for a tuple $\overline{a}_1 < s$ such that $\mathcal{A} \models \psi(v_1, \overline{a}_1, \overline{c}_i)$.
(4) If such a formula and tuple are found, attach a loop of length $3\ell + 2$ to v_1 and a loop of length $3\ell + 1$ to v_2.

These actions prevent $(\overline{c}_i, \Phi_i)$ from witnessing that $\mathcal{A}$ is relatively computably categorical: If we never find a formula φ and tuple $\overline{a}_1$, then not every singleton satisfies some $\varphi \in \Phi_i$.

If we find a formula φ and tuple $\overline{a}_1$, let s be the stage at which these are found. Then by construction, the component of v_1 at stage s embeds into the component of v_2 at stage $s+1$, and the component of v_2 at stage s embeds into the component of v_1 at stage $s + 1$. This can be extended to an embedding $\mathcal{A}_s \hookrightarrow \mathcal{A}_{s+1}$ via the identity off these components, and notably this embedding maps v_1 to v_2 and fixes $\overline{c}_i$ elementalism.

Since φ is existential, we have

$$\begin{aligned}\mathcal{A}_s \models \varphi(v_1, \overline{c}) &\implies \mathcal{A}_{s+1} \models \varphi(v_2, \overline{c})\\ &\implies \mathcal{A} \models \varphi(v_2, \overline{c}),\end{aligned}$$

but v_1 and v_2 are not automorphic.

Strategy for Meeting $\mathcal{S}_j$ (in Isolation): As the construction of $\mathcal{A}$ proceeds, we attempt to define f_j so that it maps components in $\mathcal{A}$ to components in $\mathcal{B}_j$. Finding the image of a component in $\mathcal{A}$ is a two-step process: We identify root vertices in $\mathcal{B}_j$ as those vertices having out-degree at least two (this is the sole purpose of the loops of length two). While identifying root vertices in $\mathcal{B}_j$, we also search for cycles emanating from already identified root vertices in $\mathcal{B}_j$. When we find a component in $\mathcal{B}_j$ with the same lengths of cycles emanating from it as a component in $\mathcal{A}$, we map the root vertex and cycles appropriately.

Conflicts Between Strategies and Their Resolution: Unfortunately, our action to defeat relative computable categoricity conflicts heavily with our action for computable categoricity. Trying to define a computable isomorphism between $\mathcal{A}$ and $\mathcal{B}_j$, the naive approach would be to wait for the components to appear in $\mathcal{A}$ and $\mathcal{B}_j$ and to define the isomorphism appropriately. If and when the components grow in $\mathcal{A}$ or $\mathcal{B}_j$, an opponent would have the opportunity to switch v_1 and v_2, killing our computable isomorphism f_j. As we need infinitely many pairs of components to defeat relative computable categoricity, an opponent would have sufficiently many opportunities to diagonalize against all computable functions.

The critical observation is that this opportunity to diagonalize can be prevented by slowing down the construction: For the finitely many higher-priority $\mathcal{R}_i$-strategies (which build finitely many finite components), the $\mathcal{S}_j$-strategy defines the computable isomorphism f_j nonuniformly. For the components built by lower-priority $\mathcal{R}_i$-strategies, we use the above-mentioned technique of *pushing on isomorphisms*: The $\mathcal{S}_j$-strategy will allow the lower-priority $\mathcal{R}_i$-strategy to extend its component in Step 4 only gradually as follows:

(4'a) Attach a loop of length $3\ell + 2$ to v_1.
(4'b) Wait until this loop appears in $\mathcal{B}_j$ for every $j < i$ for which $\mathcal{A} \cong \mathcal{B}_j$.
(4'c) Attach a loop of length $3\ell + 1$ to v_2.

In this way, the above problem cannot occur: At any time, we will be able to distinguish v_1 and v_2 in $\mathcal{B}_j$. Of course, it will likely be the case that $\mathcal{A} \ncong \mathcal{B}_j$ for some $j < i$, in particular that some $\mathcal{B}_j$ with $j < i$ does not have a loop of length $3\ell + 2$. Hence, we may wait at Step 4'b unnecessarily (since we cannot effectively know whether $\mathcal{A} \cong \mathcal{B}_j$), causing Step 4'c not to be reached. This would cause our diagonalization attempt against Φ_i to be unsuccessful.

The solution is to have $\mathcal{R}_i$-strategies guess the outcomes of higher priority $\mathcal{S}_j$-strategies via a priority tree. Each $\mathcal{R}_i$-strategy will have two outcomes: `wait`, (indicating that the strategy is still searching for a formula φ and a tuple $\overline{a}_1$) and `act` (indicating that the strategy has found the desired φ and $\overline{a}_1$). Each $\mathcal{S}_j$-strategy will have an infinite outcome ∞ (indicating that $\mathcal{S}_j$ believes $\mathcal{A} \cong \mathcal{B}_j$) and finite outcomes `k` for all $k \in \omega$ (counting the number of times $\mathcal{S}_j$ has taken outcome ∞).

Full Strategy for Meeting $\mathcal{R}_i$: We take the following actions, always being careful to use elements larger than those found in $\overline{c}_i$:

(1) Fix a *large* number ℓ and create two new root vertices v_1 and v_2.
(2) Attach a loop of length 2 and a loop of length 3ℓ to both v_1 and v_2 and a loop of length $3\ell + 1$ to v_1.
(3) For every formula $\varphi(x) := (\exists \overline{y})[\psi(x, \overline{y}, \overline{c}_i)]$ in Φ_i, search for a tuple $\overline{a}_1 < s$ such that $\mathcal{A} \models \psi(v_1, \overline{a}_1, \overline{c}_i)$.
(4) If such a formula and tuple are found, attach a loop of length $3\ell + 2$ to v_1.
(5) Wait until the next stage at which the strategy is accessible.
(6) Attach a loop of length $3\ell + 1$ to v_2.

While the strategy is searching at Step 3, it has outcome `wait`. Once it has found a formula φ and a tuple $\overline{a}_1$, it has outcome `act`.

Full Strategy for Meeting $\mathcal{S}_j$: Let σ on the priority tree be the $\mathcal{S}_j$-strategy in question. Let s be the current stage. Let k be the number of stages less than s at which σ had outcome ∞.

We consider certain root vertices in $\mathcal{A}$: For each $\tau \subset \sigma$ such that $\tau^\frown\texttt{wait} \subseteq \sigma$, we consider the root vertices created by τ; for each $\tau \subset \sigma$ such that $\tau^\frown\texttt{act} \subseteq \sigma$ and τ has reached Step 6, we consider the root vertices created by τ; and for each $\tau \not\subset \sigma$ with τ incomparable with $\sigma^\frown\texttt{k}$, we consider the root vertices created by τ.

For each root vertex v in $\mathcal{A}$ we are considering, if $f_j(v)$ is not yet defined, we search $\mathcal{B}_{j,s}$ for a root vertex u with a component identical to the component of v and define $f_j(v) := u$ and then extend f_j to an isomorphism of the components. If $f_j(v)$ is defined, and the component of v appears identical to the component of $f_j(v)$ in $\mathcal{B}_{j,s}$, we extend f_j to an isomorphism of the components, if it is not already.

After this action, if for every vertex v we are considering, $f_j(v)$ is defined and f_j is an isomorphism of the components of v and $f_j(v)$, then σ has outcome ∞ at stage s. Otherwise, it has outcome `k`.

Construction: We create a priority tree by devoting each level to one requirement in some effective fashion. At stage s, we let all visited strategies of length at most s act in order of priority.

Verification: Define the true path through the priority tree in the usual fashion. We note the important fact that if the current path moves to the left of a node on the priority tree that has already been visited, that node can never be visited again.

It is immediate from the construction that $\mathcal{A}$ is a computable presentation. We verify that it is both computably categorical and not relatively computably categorical.

Claim 3.1.1. The structure $\mathcal{A}$ is computably categorical.

Proof. Fix an index j such that $\mathcal{A} \cong \mathcal{B}_j$, and let σ be the $\mathcal{S}_j$-strategy along the true path. By assumption, the presentation $\mathcal{B}_j$ contains a component isomorphic to every component of $\mathcal{A}$, so σ will eventually define $f_j(v)$ for every vertex it considers. For the components built by $\tau \subset \sigma$, since σ is on the true path, these components will never grow once σ begins considering them, so f_j is correct on these.

For the components built by strategies τ incomparable with σ, τ can never be visited after σ begins considering them, and so they can never grow once they are considered. So f_j is correct on these.

For the components built by $\tau \supseteq \sigma^\frown\infty$, if τ has final outcome `wait`, then the components never grow once σ begins considering them.

If τ adds the loop of length $3\ell + 2$ to v_1, then before τ added this loop, σ defined $f_j(v_1)$ to be an element of $\mathcal{B}_j$ with a loop of size $3\ell + 1$. After τ adds this loop, σ will never again have outcome ∞ unless a loop of length $3\ell + 2$ appears attached to $f_j(v_1)$, and if σ never again has outcome ∞, then v_1 is the unique vertex with a loop of size $3\ell+1$. So the loop of length $3\ell+2$ must appear on $f_j(v_1)$.

If τ adds the loop of length $3\ell + 1$ to v_2, σ must have outcome ∞ at some stage after τ attached the loop of length $3\ell + 2$ to v_1. So $f_j(v_1)$ has a loop of size $3\ell + 2$ and one of size 3ℓ, and $f_j(v_2)$ has a loop of size 3ℓ. Then there are only two loops of size 3ℓ in $\mathcal{A}$, one with a loop of size $3\ell+2$ and one without, so by elimination $f_j(v_2)$ must be the correct image of v_2. So the loop of length $3\ell+1$ must appear on $f_j(v_2)$.

For the components built by $\tau \supseteq \sigma^\frown\mathtt{k}$ for some k, if σ is considering this component at stage s, then it has had outcome ∞ more than k many times by stage s. So the components can never again grow once they are considered, so f_j is correct on these.

By the above, since f_j is correct on every component on which it is defined, and it will be defined on every component it considers, σ must have true outcome ∞. So by construction, every component is considered, and thus f_j is an isomorphism. □

Claim 3.1.2. The structure $\mathcal{A}$ is not relatively computably categorical.

Proof. Fix an index i and let σ be the $\mathcal{R}_i$-strategy along the true path. Then either σ will wait forever at Step 3, or it will reach Step 6. In the former case, the element v_1 fails to satisfy any $\varphi \in \Phi_i$. In the latter case, the nonautomorphic elements v_1 and v_2 satisfy $\varphi \in \Phi_i$. In either case, the family Φ_i is not a Scott family. □

This concludes the proof of Theorem 3.1. □

Having illustrated the technique of pushing on isomorphisms, we return to Theorem 1.13. One might think that a simpler way to prove it would be to relativize Goncharov's Theorem 1.11. After all, if $\mathcal{A}$ is 1-decidable, then relative to $\mathbf{0}'$, the presentation $\mathcal{A}$ is 2-decidable. However, a relativized version of Goncharov's Theorem 1.11 would require a modified version of computable categoricity as a hypothesis as follows:

Corollary 3.2. *If $\mathcal{A}$ is a 1-decidable computable presentation of a structure $\mathcal{S}$ with the property that for every Δ^0_2-computable presentation $\mathcal{B}$ of $\mathcal{S}$, there is a Δ^0_2-computable isomorphism $f : \mathcal{B} \cong A$, then $\mathcal{S}$ is relatively Δ^0_2-categorical.*

We show that the hypothesis of "Δ^0_2-computable categoricity" in the above corollary is not implied by computable categoricity:

Theorem 3.3. *There is a 1-decidable, computably categorical structure $\mathcal{S}$ having a computable presentation $\mathcal{A}$ and a Δ^0_2-computable presentation $\mathcal{B}$ such that $\mathcal{A}$ and $\mathcal{B}$ are not Δ^0_2-isomorphic.*

Proof. The structure $\mathcal{S}$ is an undirected graph. If we were not seeking $\mathcal{S}$ to be computably categorical, the structure $\mathcal{S}$ could be built as the union of infinitely many substructures $\mathcal{S}_i$. Each $\mathcal{S}_i$ would consist of roots $v_{i,j}$ for $j \in \omega \cup \{\infty\}$. For $j \in \omega$, the root $v_{i,j}$ would have a loop of length p_i^{2k} for every $k < j+1$ and one of length p_i^{2j+1}; $v_{i,\infty}$ would have a loop of length p_i^{2k} for every $k \in \omega$ (here p_i is the ith prime). Thus, the substructure $\mathcal{S}_i$ would consist of an $(\omega+1)$-chain of components, with the finite components matching the infinite component for longer and longer segments, yet each having a unique loop size to distinguish it from the infinite component and other finite components.

Of course, taking $\mathcal{A}$ to be the standard presentation of $\mathcal{S}$, we could build a Δ^0_2-computable presentation $\mathcal{B}$ not isomorphic to $\mathcal{A}$ via any Δ^0_2-isomorphism. This could be done by using the substructure $\mathcal{S}_i$ to diagonalize against the ith Δ^0_2-function $\varphi_i : \mathcal{A} \to \mathcal{B}$: When φ_i converges on $v_{i,\infty}$, we make its image in $\mathcal{B}$ be $v_{i,j}$ for some large $j \in \omega$.

As we are seeking a computably categorical structure, we alter the isomorphism type of $\mathcal{S}$ to include the pushing on isomorphisms machinery. In particular, we build a computable structure $\mathcal{A}$, taking $\mathcal{S}$ to be its isomorphism type. As we are seeking a 1-decidable structure, we use *large* loop sizes rather than powers of primes. After constructing $\mathcal{A}$, we build the Δ^0_2-computable presentation $\mathcal{B}$.

The construction of the components in $\mathcal{A}$ proceeds as expected.

Construction of a Component: Using increasing numbers of loops, we build accumulation points in the Σ^c_1-type space.

(1) Set $k := 0$. Create a root vertex $v_{i,\infty}$ and attach a loop of large size $n_{i,0}$.
(2) Attach a loop of large size $n_{i,k+1}$ to $v_{i,\infty}$.
(3) Wait until the next stage this strategy is visited (this allows higher-priority isomorphism strategies to "push on isomorphisms").
(4) Create a root vertex $v_{i,k}$ with attached loops of all sizes $n_{i,0}, \ldots, n_{i,k}$. Also attach a loop of distinct large size $m_{i,k}$ to $v_{i,k}$.
(5) Increment k and return to Step 2.

Unfortunately, as described above, the resulting structure would not be 1-decidable. For example, "Does $v_{i,\infty}$ have degree at least i?" is a Σ^0_1-question, and answering it would require knowing how many times we reach Step 2. Similarly, "Are there at least i many loops of size $n_{i,0}$?" is a Σ^0_1-question that requires knowing how many times we reach Step 4. As a remedy, instead of a single root vertex $v_{i,\infty}$, we create an infinite collection of root vertices joined by infinitely many paths of length 2 (that is, we create infinitely many copies of the $v_{i,\infty}$-component, with infinitely many paths of length 2 between every two copies of the root vertex). We do the same for each root vertex $v_{i,k}$, creating an infinite collection of root vertices joined by infinitely many paths of length 2. Because of this, for any even size, there will be a loop of that size attached to $v_{i,j}$ and to $v_{i,\infty}$. So we require that our sizes $n_{i,k}$ and $m_{i,k}$ are always odd.

We create a tree of strategies as in the proof Theorem 3.1. Some levels will be devoted to S_j-strategies, which ensure that if $\mathcal{B}_j \cong \mathcal{A}$, then there is some computable isomorphism between them. Others will be devoted to R_i-strategies, which simply perform the above construction of points (with the modifications discussed).

Verification of $\mathcal{A}$: The structure $\mathcal{S}$ is computably categorical because of the pushing on isomorphism technology already illustrated: An isomorphism strategy S_j of higher priority than R_i can always distinguish the copy of $v_{i,\infty}$ in $\mathcal{B}_j$ by the $n_{i,k+1}$ loop. Because R_i-strategies wait at Step 3, no $v_{i,k}$ with this loop will be created until the copy of $v_{i,\infty}$ in $\mathcal{B}_j$ has distinguished itself with a larger loop. Lower-priority S_j-strategies non-uniformly know the image of $v_{i,\infty}$ in $\mathcal{B}_j$. The image of $v_{i,k}$ can always be uniquely identified by the loop of size $m_{i,k}$.

Of course, the above is not quite correct, because we create infinite collections of each $v_{i,\infty}$ and each $v_{i,k}$. So rather than uniquely identifying the point $v_{i,\infty}$ or $v_{i,k}$ in $\mathcal{B}_j$, we uniquely identify the collection. Once the collection has been found, however, a simple back-and-forth construction can construct the isomorphism.

Claim 3.3.1. The presentation $\mathcal{A}$ is 1-decidable.

Proof. It suffices to show that for any canonically given finite graph G, we can effectively determine whether or not G occurs as an induced subgraph of $\mathcal{A}$. For a canonically given finite graph G, we wait until a stage s in the construction when a loop of some size $n_{i,k} > |G|$ has been enumerated into the construction, and then we answer whether or not G is an induced subgraph of $\mathcal{A}$ as follows.

First, we identify all simple loops in G of odd length. If any of these loops have more than 1 vertex of degree greater than 2, we know that G is not an induced subgraph of $\mathcal{A}$. If any of these loops are of a size we have not yet used as some $n_{i,k}$ or $m_{i,k}$ by stage s, then since our loop sizes are always chosen large, no loop of that size will ever be used, and so we know G is not an induced subgraph of $\mathcal{A}$.

Otherwise, every simple loop of odd length has size some $n_{i,k}$ or $m_{i,k}$ already chosen during the construction. If some loop of size $n_{i,k}$ and some other loop of size $n_{i',k'}$ with $i \neq i'$ are in the same component, then we know G is not an induced subgraph of $\mathcal{A}$. Similarly, if a loop of size $n_{i,k}$ is in the same component as a loop of size $m_{i',k'}$ with $i \neq i'$ or $k > k'$, then we know that G is not an induced subgraph of $\mathcal{A}$. Also, if a loop of size $m_{i,k}$ is in the same component as a loop of size $m_{i',k'}$ with $i \neq i'$ or $k \neq k'$, then we know that G is not an induced subgraph of $\mathcal{A}$. Finally, if two distinct simple loops of odd length intersect, then we know that G is not an induced subgraph of $\mathcal{A}$.

Otherwise, call a vertex in G a *root* if it has degree greater than 2. Note that any embedding of G as an induced subgraph of $\mathcal{A}$ must map the roots of G to roots of $\mathcal{A}$. Let G' be the induced subgraph of G containing those vertices which are roots and also those vertices which are not part of a simple loop of odd length. Any embedding of G as an induced subgraph of $\mathcal{A}$ will give a two-coloring of G' which colors all the roots of G' red and such that every vertex colored blue has degree at most 2: Color a vertex red if it maps to a root of $\mathcal{A}$, and blue otherwise.

Conversely, if G' admits a two-coloring which colors all its roots red and such that every vertex colored blue has degree at most 2, then G can be embedded into $\mathcal{A}$ as an induced subgraph: For each component, if it contains a loop of size $m_{i,k}$, then map that component into the collection of copies of $v_{i,k}$, mapping the red vertices to roots in $\mathcal{A}$; it the component does not contain a loop of size $m_{i,k}$ for any k, but does contain one of size $n_{i,k}$ for some k, map the component into the collection of copies of $v_{i,\infty}$, again mapping red vertices to roots in $\mathcal{A}$; if the component contains no simple loops of odd sizes, then map the component into the collection of copies of $v_{0,\infty}$.

Thus we can decide if G is an induced subgraph of $\mathcal{A}$ by considering the finitely many two-colorings of G'. $\square$

Construction of $\mathcal{B}$: We work in the presence of a $\mathbf{0}'$-oracle. We begin by simply copying $\mathcal{A}$, while simultaneously studying Δ^0_2-functions from $\mathcal{A}$ to $\mathcal{B}$. When a

Δ^0_2-function φ_ℓ converges on some accumulation point $v_{i,\infty} \in \mathcal{A}$ with $i > \ell$, we may assume $v'_{i,\infty} := \varphi_\ell(v_{i,\infty})$ is a copy of $v_{i,\infty}$ in $\mathcal{B}$ (as otherwise we have won against φ_ℓ). We use our oracle to determine if R_i will ever again reach Step 2 and then Step 4. If so, we pause the construction of $v'_{i,\infty}$ until this happens. We make $v'_{i,\infty}$ the image of the new $v_{i,k}$ instead of $v_{i,\infty}$, defeating the function φ_ℓ. Since we are requiring that $i > \ell$, our approximation to $\varphi_\ell(v_{i,\infty}) \in \mathcal{B}$ reaches a limit. □

Just as we relativized Goncharov's result to $\mathbf{0}'$ to weaken the decidability requirement, we can do the same for our Theorem 1.13:

Corollary 3.4. *If a computable structure $\mathcal{A}$ is such that every Δ^0_2-computable copy is isomorphic via a Δ^0_2-computable isomorphism, then $\mathcal{A}$ is relatively Δ^0_3-categorical.*

We have already seen that the hypothesis of "Δ^0_2-computable categoricity" in the above corollary is not implied by computable categoricity. Here we show that the implication can fail very badly:

Theorem 3.5. *There is a computably categorical structure $\mathcal{A}$ such that every noncomputable Δ^0_2-degree computes a presentation $\mathcal{B}$ not isomorphic to $\mathcal{A}$ by any Δ^0_2-isomorphism.*

Proof. Our structure is a directed graph consisting of pairs of components. Each pair will contain a larger component and a smaller component and be assigned a distinct prime p. The larger component will be a vertex with loops of sizes p^k for all $k \leq r+2$, while the smaller component will be a vertex with loops of sizes p^k for all $k \leq r$. The parameter r will initially be 1, and will grow (possibly to infinity) as the construction proceeds.

Let $\{\mathcal{M}_e\}_{e\in\omega}$ be an enumeration of all partial computable structures, $\{X_i\}_{i\in\omega}$ an enumeration (of partial characteristic functions) of all Δ^0_2-sets, and $\{g_j\}_{j\in\omega}$ an enumeration of all partial Δ^0_2-functions. We meet the following requirements:

$$\begin{aligned}
\mathcal{N}_e: &\quad \mathcal{M}_e \cong \mathcal{A} \Rightarrow \exists f \leq_T \emptyset\ [f : \mathcal{M}_e \cong \mathcal{A}]\\
\mathcal{R}_i: &\quad \mathcal{B}_i \leq_T X_i \text{ and } \mathcal{B}_i \cong \mathcal{A}\\
\mathcal{P}_{i,j}: &\quad X_i >_T 0 \Rightarrow \neg[g_j : \mathcal{A} \cong \mathcal{B}_i]
\end{aligned}$$

Strategy for meeting $\mathcal{N}_e$: This is a standard pushing on isomorphisms strategy.

Strategy for meeting $\mathcal{R}_i$: $\mathcal{R}_i$ maintains a Turing functional Γ_i with $\mathcal{B}_i = \Gamma_i^{X_i}$ and a bijection F_i mapping components of $\mathcal{A}$ to components of $\mathcal{B}_i$. At every stage, $\mathcal{R}_i$ grows the components of $\mathcal{B}_i$ to match the corresponding components in $\mathcal{A}$. These facts about new loops in $\mathcal{B}_i$ are enumerated into Γ_i with use k where p^k is the size of the loop.

If X_i changes to a new version, removing certain loops from $\mathcal{B}_i$, we restore those loops to $\mathcal{B}_i$ by enumerating new axioms for them into Γ_i. The exception is if $g_{j,s}(x)\!\downarrow= F_{i,s}(x)$ for some root vertex x of the larger component of some $\mathcal{P}_{i,j}$-strategy, and the largest two loops attached to $F_{i,s}(x)$ are removed: then we instead take the opportunity to redefine $F_i(x)$, interchanging the role of larger and smaller components in $\mathcal{B}_i$.

Strategy for meeting $\mathcal{P}_{i,j}$: We initially choose an unused large prime p and begin building (in $\mathcal{A}$) the components for p. Let $x \in \mathcal{A}$ be the root vertex of the larger component. The behavior of the strategy at stage s depends on whether $g_{j,s}(x)\!\downarrow=$

$F_{i,s}(x)$. If so, we increment r, adding a new loop to each of the two components. If not, we do nothing.

Construction: We place the $\mathcal{N}_e$- and $\mathcal{P}_{i,j}$-strategies on a priority tree in the standard fashion. The $\mathcal{R}_i$-strategies are not placed on the tree, but instead act at every stage.

Verification: Clearly $\mathcal{A}$ is a total computable structure.

Claim 3.5.1. The $\mathcal{N}_e$-strategies ensure their requirement.

Proof. This is the now-familiar "pushing on isomorphisms" argument: If $\mathcal{N}_e$ is of higher priority than a pair being constructed, then the pair respects $\mathcal{N}_e$'s isomorphism by only adding one loop at a time. If $\mathcal{N}_e$ is of lower priority than a pair of components, then it nonuniformly knows which component is larger and which is smaller. □

Claim 3.5.2. The $\mathcal{R}_i$-strategies ensure their requirement.

Proof. If X_i is not a true Δ^0_2-set, then $\mathcal{R}_i$ is trivially satisfied, so we assume that it is. The use of the loops in $\mathcal{B}_i$ does not grow, so since X_i eventually stops changing on initial segments, $\mathcal{B}_i$ eventually stops changing. Thus $\mathcal{B}_i$ is an X_i-computable structure.

Furthermore, at every stage s, $F_{i,s} : \mathcal{A}_s \to \mathcal{B}_{i,s}$ is an isomorphism. Thus on all components where $F_i = \lim_s F_{i,s}$ exists, $\mathcal{A}$ is isomorphic to $\mathcal{B}_i$. The only components at which this limit might not exist are components built by $\mathcal{P}_{i,j}$-strategies that infinitely often see $g_{j,s}(x) = F_{i,s}(x)$. But such components have their r grow to infinity, and thus the larger and smaller components are identical. Thus it does not matter which component maps to which, and we may extend F_i to an isomorphism $\mathcal{A} \cong \mathcal{B}_i$. □

Note that we cannot ask that $F_i = \lim_s F_{i,s}$ be a total isomorphism, because then we would be unable to meet requirement $\mathcal{P}_{i,j}$ with $g_j = F_i$.

Claim 3.5.3. The $\mathcal{P}_{i,j}$-strategies ensure their requirement.

Proof. Assume $X_i >_T \emptyset$ is a Δ^0_2-set. Let x be the root vertex of the larger component built by the $\mathcal{P}_{i,j}$-strategy along the true path. If $g_j(x) = \lim_s g_{j,s}(x)$ does not exist, the requirement is trivially satisfied, so assume $g_j(x)$ exists.

By standard Δ^0_2-permitting, there is some stage after $g_j(x)$ has converged at which the $\mathcal{R}_i$-strategy has an opportunity to redefine $F_i(x)$, and this change is not permanently reverted by X_i returning to an earlier configuration. So by construction, $F_i(x) \neq g_j(x)$, and the components built by the $\mathcal{P}_{i,j}$-strategy are finite. Thus g_j cannot be an isomorphism. □

This completes the proof of Theorem 3.5. □

4. A Computably Categorical Structure Which is Not Relatively Arithmetically Categorical

In this section, we prove our main result:

Theorem 1.14. *There is a computably categorical structure which is not relatively arithmetically categorical.*

First, however, we need the previously mentioned fact that relative arithmetic categoricity is equivalent to relative Δ^0_n-categoricity for some $n \in \omega$.

Proposition 1.7. *A structure is relatively arithmetically categorical if and only if it is relatively Δ^0_n-categorical for some $n < \omega$.*

A structure is relatively hyperarithmetically categorical if and only if it is relatively Δ^0_α-categorical for some $\alpha < \omega_1^{CK}$.

Proof (Sketch). Fix a computable presentation $\mathcal{A}$ of a relatively arithmetically categorical structure. The key observation is that the proof of Theorem 1.9 (as presented in Theorem 10.14 of [2]) does not use the existence of an isomorphism $\pi : \mathcal{B} \cong \mathcal{A}$ arithmetic in $\mathcal{B}$ for every presentation $\mathcal{B}$ isomorphic to $\mathcal{A}$, but instead constructs a particular generic presentation $\mathcal{B}_0$ and only uses the existence of such an isomorphism for this fixed presentation.

For this fixed presentation $\mathcal{B}_0$, there is an isomorphism $\pi : \mathcal{B}_0 \cong \mathcal{A}$ arithmetic in $\mathcal{B}_0$. Thus, there is a $\Delta^0_n(\mathcal{B}_0)$-isomorphism for some $n < \omega$. From this, it follows by the proof of Theorem 1.9 that there is a c.e. Scott family of Σ^c_n-formulas for the structure, and thus the structure is relatively Δ^0_n-categorical.

A similar observation applies to $\mathcal{A}$ being relatively hyperarithmetically categorical. □

We also set some notation on strings.

Definition 4.1. Fix a tree $T \subseteq \omega^{<\omega}$. The *cone in T above a string σ*, denoted as $[\sigma]$, is the set of strings $\{\tau \in T : \sigma \subseteq \tau\}$. Strings $\sigma, \tau \in T$ are *incomparable*, denoted as $\sigma \mid \tau$, if neither $\sigma \subseteq \tau$ nor $\tau \subseteq \sigma$.

Proof of Theorem 1.14. The structure $\mathcal{A}$ we construct is a directed graph. In order to facilitate its construction, we build, for each $n \in \omega$, a c.e. tree $T_n \subset \omega^{\leq n}$ of height n external to the structure $\mathcal{A}$. We effectively embed T_n (as a directed graph) into $\mathcal{A}$, denoting the image of a string $\sigma \in T_n$ by z_σ. We also attach (finitely or infinitely many) cycles of various lengths to each vertex z_σ. The structure $\mathcal{A}$ will consist entirely of the embedded trees T_n and the cycles attached to these vertices.

We will construct the trees T_n so that, for those that are infinite, there will be infinitely many $j \in \omega$ such that $\langle j \rangle$ has infinitely many extensions in T_n. Moreover, if $\langle j \rangle$ and $\langle j' \rangle$ both have infinitely many extensions in T_n, then it will be the case that the parameter-free Σ^c_{n-2}-type of $z_{\langle j \rangle}$ will equal the parameter-free Σ^c_{n-2}-type of $z_{\langle j' \rangle}$. As $z_{\langle j \rangle}$ and $z_{\langle j' \rangle}$ will not be automorphic in $\mathcal{A}$, this will ensure that there is no parameter-free Σ^c_{n-2}-formula which isolates the type of $z_{\langle j \rangle}$.

Terminology: We identify elements in $\mathcal{A}$ with their transitive closures (under the "forward" direction of the directed graph relation). Thus we say an element is *infinite* if its transitive closure is infinite, we say two elements are *isomorphic* if their transitive closures are isomorphic, and so on.

Our construction will use three types of devices to mark elements: Leaf labels, height labels, and temporary labels. Each type of device will make use of cycles (loops), but we will partition ω so that no cycle length is used by two different sorts of devices.

- *Leaf labels* will mark the "leaves" of the image of T_n.
- *Height labels* will mark elements in the image of T_n as coming from strings of length m in T_n.

- *Temporary labels* will be a collection (possibly finite, possibly infinite) of loops, at most one of each finite size. At any finite stage, the largest loop in a given temporary label will not occur in any other temporary label, making distinct temporary labels non-isomorphic at this stage. However, if two temporary labels both grow to be infinite, then they will be isomorphic in the limit since they will have the same collection of loops. (The above is not quite correct, since some temporary labels will deliberately be made identical to others, but is true of all temporary labels which are not deliberately identical.)

As already indicated, for $\sigma \in T_n$, we let z_σ denote the corresponding element in $\mathcal{A}$. Though there is ambiguity with this notation (as a string σ may belong to various trees T_n for varying n), it will always be clear which tree T_n is under discussion. Thus when we refer to z_σ, we refer to the image of the string σ in T_n within $\mathcal{A}$.

For $\sigma \in T_n$, we let the *retinue* of σ be the set of vertices in $\mathcal{A}$ consisting of z_σ itself, the vertices in the temporary or height labels for z_σ, as well as those vertices in the leaf labels attached to z_σ (if σ is a leaf in T_n). Let the retinue of T_n be the union of the retinues of its elements.

Basic Structure of a Tree T_n: The constructions of the trees T_n are mostly independent, so we focus on a single n. The tree T_n will be a c.e. subtree of the tree[2]

$$\hat{T}_n := \left\{\sigma \in \omega^{\leq n} \mid \sigma \neq \emptyset \text{ and } (\forall m < |\sigma| - 1)\ [\sigma(m) \neq \sigma(m+1)]\right\}$$

The tree T_n will be a proper subtree (or rather, subforest) of $\hat{T}_n$ satisfying the following properties:

(I) The element $\langle 0\rangle$ and infinitely many elements $\langle j\rangle$ (for $j > 0$) are in T_n and are infinite. Let J_n be the set of all $j \in \omega$ such that $\langle j\rangle$ is infinite.

(II) $\hat{T}_n \cap J_n^{\leq n} \subseteq T_n$.

(III) If $\sigma^\frown\langle j\rangle, \sigma^\frown\langle k\rangle \in \hat{T}_n \cap J_n^{\leq n-1}$, then

$$[\sigma^\frown\langle j\rangle] - [\sigma^\frown\langle j\rangle^\frown\langle k\rangle] \cong [\sigma^\frown\langle k\rangle] - [\sigma^\frown\langle k\rangle^\frown\langle j\rangle]$$

via the map $\sigma^\frown\langle j\rangle^\frown\rho \mapsto \sigma^\frown\langle k\rangle^\frown\rho$.

(IV) The isomorphism in (III) extends to an isomorphism of the retinues.

These properties will ensure that the parameter-free Σ^c_{n-2}-type of $z_{\langle j\rangle}$ equals the parameter-free Σ^c_{n-2}-type of $z_{\langle j'\rangle}$ for any $j, j' \in J_n$. Because of the priority construction later, only infinitely many of the trees will satisfy the above, but that will suffice.

Basic Structure of Labels: In order to ensure our marking devices do not interfere with each other, we effectively partition $\omega - \{0, 1\}$ into disjoint infinite computable sets W^L, W^H, and W^T. We further effectively partition W^T into disjoint infinite computable sets $\{W^{\langle j,\alpha\rangle}\}_{j\in\omega,\alpha\in T}$, where T is the priority tree detailed in the upcoming section "Strategies, Outcomes, and the Priority Tree".

[2] Technically, the set of vertices $\hat{T}_n$ forms a forest rather than a tree since $\hat{T}_n$ does not contain the empty string $\emptyset$. Throughout this proof, we abuse terminology and refer to such sets as *trees* rather than forests.

We effectively enumerate the set W^L as $W^L := \{w_0^L < w_1^L < \dots\}$. If $\sigma \in T_n$ is a leaf (this will be the case if and only if $|\sigma| = n$), we will attach a leaf label of size $w^L_{\sigma(n-1)}$ to z_σ.

We effectively enumerate the set W^H as $W^H := \{w_0^H < w_1^H < \dots\}$. If $\sigma \in T_n$ has height m, we will attach a height label of size $w^H_{\langle m,n\rangle}$ to z_σ.

We effectively enumerate the sets $W^{\langle j,\alpha\rangle}$, for each $j \in \omega$ and $\alpha \in T$ (where again T is the priority tree detailed later), as $W^{\langle j,\alpha\rangle} := \{w_0^{\langle j,\alpha\rangle} < w_1^{\langle j,\alpha\rangle} < \dots\}$. A temporary label will have a *type* $\langle j,\alpha\rangle$, where $j \in \omega$ and $\alpha \in T$. There will be a single c.e. set S shared by all temporary labels throughout the construction. Let S initially be empty.

A temporary label of type $\langle j,\alpha\rangle$ is built at an element z by performing the following steps, starting with $k = 0$:

(1) Attach a loop of size $w_0^{\langle j,\alpha\rangle}$ to z.
(2) Attach a loop of size $w_{k+1}^{\langle j,\alpha\rangle}$ to z.
(3) Enumerate $w_k^{\langle j,\alpha\rangle}$ into S.
(4) For every m in S with $m < w_k^{\langle j,\alpha\rangle}$, if z does not already have a loop of length m attached, add one.
(5) Increment k and return to Step 2.

We only perform a step of this construction when the strategies described below indicate we should. Clearly, if an instance of a label acts infinitely many times, it will have one loop of size m for every $m \in S$ and nothing else. However, at every finite stage, the label will be distinguished by the loop of size $w_{k+1}^{\langle j,\alpha\rangle}$.

Before continuing, we note an important consequence of these marking devices. In any computable presentation $\mathcal{B}$ isomorphic to $\mathcal{A}$, the set of vertices that are images of the trees T_n is computable. This is because, given a vertex $x \in A$, either x has out-degree one and is part of a cycle with another vertex of the cycle having out-degree at least two (in which case the vertex with out-degree at least two is z_σ for some σ and x is in the retinue of σ); or x has out-degree at least two, including a cycle of some length $w^H_{\langle m,n\rangle}$ (in which case x is in the image of T_n).

Moreover, given such a vertex $z_\sigma \in B$, both n and $|\sigma|$ can be effectively determined from the length of the cycle $w^H_{\langle m,n\rangle}$.

Requirements: Let $\{\mathcal{B}_\ell\}_{\ell\in\omega}$ be an effective enumeration of all (partial) computable directed graphs. We have two sorts of requirements, for all $n, \ell \in \omega$:

Υ_n : The tree T_n satisfies properties (I), (II), (III), and (IV) above.
Φ_ℓ : If $\mathcal{B}_\ell \cong \mathcal{A}$, then $f_\ell : \mathcal{B}_\ell \cong \mathcal{A}$ for some computable function f_ℓ.

We will meet Φ_ℓ for every ℓ, but will only meet infinitely many of the Υ_n.

Strategies, Outcomes, and the Priority Tree: The above requirements necessitate the following three types of strategies:

Υ_n : These strategies are the main strategies for the Υ_n-requirement. Each will try to build an infinite tree T_n satisfying (I), (II), (III), and (IV). This strategy will have only one outcome.
$\Xi_{n,j}$: To each Υ_n-strategy, we attach a $\Xi_{n,j}$-strategy for each $j \in \omega$, which will try to place j into J_n, and if $j \in J_n$, will help ensure that T_n satisfies the above properties. This strategy will have only one outcome.

- Φ_ℓ : This strategy will build a partial computable function f_ℓ so that $f_\ell : \mathcal{B}_\ell \cong \mathcal{A}$ if $\mathcal{B}_\ell \cong \mathcal{A}$. It does so by utilizing the pushing on isomorphisms machinery: By slowing down the construction below its "isomorphic" outcome ∞ and tagging all elements created there with temporary tags, by permanently tagging all elements created by strategies to the right of its "isomorphic" outcome ∞, and by nonuniformly building f_ℓ for the finitely many elements created by strategies above and to the left of the strategy. In addition to its "isomorphic" outcome ∞, a Φ_ℓ-strategy will also have infinitely many finitary outcomes k (for $k \in \omega$), denoting that the Φ_ℓ-strategy has k many times taken the "isomorphic" outcome ∞.

We now effectively assign strategies to nodes of a *priority tree* $T \subset (\{\infty\} \cup \omega)^{<\omega}$ (not to be confused with the trees T_n for $n \in \omega$) with the following properties:

- Each level of the tree is entirely devoted to either Υ-, Ξ- or Φ-requirements;
- If a level is devoted to Φ-requirements, every node on that level is assigned to the same Φ_ℓ-requirement.
- For every ℓ, there is a level devoted to the Φ_ℓ-requirement;
- For every n and every j, there is precisely one node devoted to the Υ_n-requirement and precisely one node devoted to the $\Xi_{n,j}$-requirement;
- There are infinitely many levels devoted to Υ-requirements;
- For every n, j, the $\Xi_{n,j}$-node is an extension of the Υ_n-node;
- For every Υ_n-node $\alpha \in T$, there are infinitely many levels such that every node on that level which extends α is a $\Xi_{n,j}$-node for some j;
- A Φ_ℓ-strategy $\alpha \in T$ has infinitely many immediate successors on T, namely, $\alpha^\frown\langle\infty\rangle$ and $\alpha^\frown\langle k\rangle$ for all $k \in \omega$, ordered $\alpha^\frown\langle\infty\rangle <_T \cdots <_T \alpha^\frown\langle 1\rangle <_T \alpha^\frown\langle 0\rangle$;
- An Υ_n- or $\Xi_{n,j}$-node $\alpha \in T$ has only one immediate successor on T, namely, $\alpha^\frown\langle\infty\rangle$; and
- If $\alpha \in T$ is a Υ_n-node, $\alpha^\frown\langle\infty\rangle$ is the $\Xi_{n,0}$-node.

Note that our construction will have the feature that if a node is visited, and then later a node to its left is visited, the original node will never be visited again.

The Υ_n-Strategy: This strategy merely serves as the top of the cone of strategies on T building T_n. It performs no action and always takes the outcome ∞.

The $\Xi_{n,j}$-Strategy: For a $\Xi_{n,j}$-strategy α, let

$$J_\alpha := \{j' \mid \text{some } \gamma \subseteq \alpha \text{ is a } \Xi_{n,j'}\text{-strategy}\}.$$

Note that J_α is finite and uniformly computable. The strategy α believes that $J_\alpha \subset J_n$, so α works to ensure the properties of T_n within $\hat{T}_n \cap J_\alpha^{\leq n}$.

Let

$$R_\alpha := \{\sigma \in \hat{T}_n \cap J_\alpha^{\leq n} \mid \sigma(k) = j \text{ for some } k\}.$$

The strategy α believes that the strings in R_α should be infinitely branching in T_n. Furthermore, no strategy $\alpha' \subset \alpha$ believes any string in R_α should be infinitely branching in T_n (since $j \notin J_{\alpha'}$).

The strategy α acts as follows:

(1) When α is first visited, it enumerates all of R_α into T_n. It also creates corresponding elements $z_\sigma \in A$ for each $\sigma \in R_\alpha$. Each new element z_σ is

given a height label of size $w^H_{\langle|\sigma|,n\rangle}$ marking its height and the beginnings of a temporary label of type $\langle\sigma(|\sigma|-1),\alpha\rangle$.

If $|\sigma| = n$, then z_σ is also given a leaf label of size $w^L_{\sigma(n-1)}$.

(2) Whenever α is visited, for every pair of strings (σ,ρ) with $\sigma \in J^{<n}_\alpha$ and $|\rho| > 0$, and for all $k, k' \in J_\alpha$, if $\sigma^\frown\langle k\rangle^\frown\rho \in T_n$ and $\sigma^\frown\langle k'\rangle^\frown\rho \in \hat{T}_n \backslash T_n$, then α enumerates $\sigma^\frown\langle k'\rangle^\frown\rho$ into T_n. It also creates a corresponding element $z_{\sigma^\frown\langle k'\rangle^\frown\rho}$ in A with the appropriate height label and leaf label (if appropriate) and a temporary label created by copying the entirety of the temporary label of $z_{\sigma^\frown\langle k\rangle^\frown\rho}$.

(3) Whenever α is visited, for every string $\sigma \in R_\alpha$, one step is taken in the construction of the temporary label of type $\langle j,\alpha\rangle$ for z_σ.

The Φ_ℓ-strategy: This is the only type of strategy with more than one outcome. It will have a Π^0_2-outcome ∞ corresponding to the isomorphic outcome and infinitely many finitary outcomes k (for $k \in \omega$), to be discussed later. As we are only measuring length of agreement on some parts of the structures, it is possible for the true outcome of a Φ_ℓ-strategy to be ∞ even if $\mathcal{B}_\ell \ncong \mathcal{A}$.

Suppose $\mathcal{B}_\ell$ is isomorphic to $\mathcal{A}$. We would like to construct a computable isomorphism between them by running a back-and-forth argument, but in order for this argument to succeed, we need to know more about $\mathcal{B}_\ell$. Suppose that, in a computable fashion, we could identify each $z_\sigma \in \mathcal{B}_\ell$. Then we would be able to construct a computable isomorphism. The job of a Φ_ℓ-strategy α will be to tag the elements of $\mathcal{B}_\ell$ with this information for all but finitely many σ, with the remainder being handled non-uniformly.

In what follows, we restrict our attention to the image of a tree T_n and all the retinues of its elements. Note that the sets R_β for $\beta \subset \alpha$ are finite and computable uniformly in α. If α has true outcome ∞, then α must tag every element z_σ except for the finitely many σ in these R_β.

Suppose that α has sufficiently high priority to halt the growth of the temporary label attached to $z_{\langle j\rangle}$. Then whatever element in $\mathcal{B}_\ell$ is tagged as $z_{\langle j\rangle}$, α can enforce this map to be correct: Let β be the strategy which enumerated $\langle j\rangle$ into T_n. Then the temporary label attached to $z_{\langle j\rangle}$ is the unique label attached to a vertex of height 1 and containing a loop of size $w^{\langle j,\beta\rangle}_k$. Suppose the element of $\mathcal{B}_\ell$ tagged as $z_{\langle j\rangle}$ also contains a loop of size $w^{\langle j,\beta\rangle}_k$. Suppose the temporary label construction performs Step (2) (attaching a loop of size $w^{\langle j,\beta\rangle}_{k+1}$). Then α will not permit the temporary label to perform Step (3) until such time as the element of $\mathcal{B}_\ell$ which is tagged as $z_{\langle j\rangle}$ gains a loop of size $w^{\langle j,\beta\rangle}_{k+1}$ in its temporary label. If this never happens, then necessarily $\mathcal{A} \ncong \mathcal{B}_\ell$. In this way, α maintains the isomorphism.

In general, if there is no $\tau \subseteq \sigma$ with $\tau \in R_\beta$ for some $\beta \subset \alpha$, then α has sufficient priority to halt the growth of the temporary label attached to z_τ for every $\tau \subseteq \sigma$ and thus can inductively enforce that the element it tags as z_τ is correct.

On the other hand, suppose for some $\tau \subseteq \sigma$, $\tau \in R_\beta$ for some $\beta \subset \alpha$. Then we cannot hope to computably tag z_σ, as identifying z_σ would entail identifying z_τ; simply search for the predecessor of z_σ permanently labeled as having height $|\tau|$. And we cannot computably tag z_τ, as α does not have sufficiently high priority to enforce this map.

For this reason, we will employ two sorts of tags: *provisional* and *non-provisional.* Non-provisional tags will be tags for which we can enforce correctness and will consequently never need to move. Non-provisional tags are of the form z_σ, for some $\sigma \in T_n$. They indicate that the tagged element is the image of z_σ in $\mathcal{B}_\ell$.

Provisional tags, on the other hand, may need to move if we have incorrectly guessed the immediate predecessor of z_σ. If we have guessed correctly, they will not move. Provisional tags are of the form $z_{b,\rho}$. They indicate that we currently guess that the tagged element is the image in $\mathcal{B}_\ell$ of $z_{\tau^\frown\rho}$ for some τ of length b which we do not have sufficiently high priority to control. There will be multiple instances of every provisional tag, since there are multiple strings τ of length b for which we lack the priority to control; we will then later, nonuniformly, determine the correct images of these z_τ, and then, uniformly from this information, the images of the $z_{\tau^\frown\rho}$.

Let k be the number of times α has taken outcome ∞. Let

$$I_\alpha := \bigcup_{\beta | \alpha^\frown\langle k\rangle} J_\beta.$$

We restrict our attention to $\sigma \in T_n \cap I_\alpha^{\leq n}$ with $\sigma \notin R_\beta$ for any $\beta \subset \alpha$. Let m be size of the largest loop in the temporary label of z_σ. There are four cases.

(1) If $|\sigma| = 1$, then we search for an element having a height label of size $w^H_{1,n}$ and with an attached temporary label containing a loop of size m. This element is non-provisionally tagged as z_σ.

(2) If $|\sigma| > 1$ and for every $\sigma' \subseteq \sigma$, $\sigma' \notin R_\beta$ for any $\beta \subset \alpha$, then we wait until some element of the image of T_n in $\mathcal{B}_\ell$ is non-provisionally tagged as z_{σ^-}. We then search among the immediate successors of this element for an element with an attached temporary label containing a loop of size m. This immediate successor is non-provisionally tagged as z_σ.

(3) If $\sigma^- \in R_\beta$ for some $\beta \subset \alpha$, then $\sigma = \sigma^{-\frown}\langle j\rangle$. There are a finite number of strings ρ with $|\rho| = |\sigma^-|$ and $\rho \in R_\gamma$ for some $\gamma \subset \alpha$. Let d be the number of such ρ.

We search for d many distinct elements in the image of T_n in $\mathcal{B}_\ell$ with a height label of size $w^H_{|\sigma^-|,n}$. For each such element, we wait until either a child appears with a loop of size m in the attached temporary label or we decide that we are in the wrong location. In the former case, we provisionally tag the child as $z_{|\sigma^-|,\langle j\rangle}$. In the latter case, we search for the next element with a height label of size $w^H_{|\sigma^-|,n}$ and try again. Even once we have provisionally tagged an element, we may decide that we are in the wrong location, in which case we remove the provisional tag and try again.

There are two reasons why we might decide an element b is the wrong location (here b is the element of height $|\sigma^-|$): b or some predecessor of b might be tagged with a provisional tag; or some element in the same connected component as b might be tagged with a non-provisional tag.

(4) Otherwise, let $\sigma = \sigma'^\frown\tau$ with σ' maximal such that $\sigma' \in R_\beta$ for some $\beta \subset \alpha$ (note that $|\tau| > 1$). We again let d be the number of ρ with $|\rho| = |\sigma'|$ and $\rho \in R_\gamma$ for some $\gamma \subset \alpha$. We wait for d many elements with a height label of size $w^H_{|\sigma^-|,n}$ to become provisionally tagged with $z_{|\sigma'|,\tau^-}$, and for each we search for a child of that element with a loop of size m in the attached temporary label. We wait until either we find such an element or we decide that the tag for $z_{|\sigma'|,\tau^-}$ was in the wrong location. In the former case, we provisionally tag the child with $z_{|\sigma'|,\tau}$. In the latter case, we try again.

> Even once we have provisionally tagged an element, we may decide that we are in the wrong location, in which case we remove the provisional tag and try again.

At a stage t, if for all n the strategy α has dispensed a tag (provisional or non-provisional) for every $\sigma \in T_n \cap I_\alpha^{\leq n}$ with $\sigma \notin R_\beta$ for any $\beta \subset \alpha$, and the retinues of each of these tagged elements appear to be isomorphic to the retinues of the corresponding z_σ, then α has outcome ∞. Otherwise, it has outcome k, the number of times α previously had outcome ∞.

Verification: Define the true path in the usual fashion. We verify that the structure $\mathcal{A}$ has the desired properties in a number of claims. The claims below verify that the tree T_n is as advertised (Claims 4.2 through 4.22), the non-relative arithmetic categoricity of $\mathcal{A}$ (Claims 4.23 through 4.27), and the computable categoricity of $\mathcal{A}$ (Claims 4.28 through 4.38).

Claim 4.2. *The set T_n is a tree, i.e., it is downwards closed except for the empty string. Moreover, it is a subtree of $\hat{T}_n$.*

Proof. We claim that T_n is downwards closed at every stage, proceeding by induction on stages.

If $\sigma \in R_\alpha$ and $|\sigma| > 1$, then either $\sigma^- \in R_\alpha$ or $\sigma^- \in R_\beta$ for some $\beta \subset \alpha$. In either case, when α is reached, the string σ^- has been enumerated into T_n.

If $\sigma^\frown\langle k'\rangle^\frown\rho$ is enumerated into T_n because $\sigma^\frown\langle k\rangle^\frown\rho \in T_n$, then by hypothesis $\sigma^\frown\langle k\rangle^\frown\rho^- \in T_n$, and thus $\sigma^\frown\langle k'\rangle^\frown\rho^-$ is enumerated into T_n if it was not already.

Strings only enter T_n by the action of some $\Xi_{n,j}$-strategy. These strategies are careful not to enumerate the empty string, a string with consecutive identical numbers in the string, or a string of length greater than n. □

Claim 4.3. *Fix a Υ_n-strategy α along the true path. Let*

$$J_n' := \{j \geq 0 \mid \textit{the } \Xi_{n,j}\textit{-strategy is along the true path}\}.$$

Then $J_n' = J_n$ (where J_n was defined in (I) *of the "Basic Structure" section).*

Proof. We have $0 \in J_n'$ since the $\Xi_{n,0}$-strategy attached to α is along the true path if and only if α is. We therefore assume $j > 0$, showing $j \in J_n'$ implies $j \in J_n$, and $j \notin J_n'$ implies $j \notin J_n$.

Suppose $j \in J_n'$. Then let β be the $\Xi_{n,j}$-strategy. Since β is on the true path, there are arbitrarily many k such that a $\Xi_{n,k}$-strategy $\gamma \supset \beta$ is visited during the construction. The first time γ is visited, the string $\langle j, k\rangle$ is enumerated into T_n. Thus $j \in J_n$.

Suppose $j \notin J_n'$. Then let β be the $\Xi_{n,j}$-strategy. Since β is not on the true path, there is some stage s_0 after which β is never again visited. After s_0, no extension of $\langle j\rangle$ will be in R_γ for any γ visited, and thus no extension of $\langle j\rangle$ will be enumerated into T_n. Thus the extensions of $\langle j\rangle$ in T_n are precisely the finitely many it possessed at stage s_0. So $j \notin J_n$. □

Claim 4.4. *Fix a Υ_n-strategy α along the true path. Then $\hat{T}_n \cap J_n^{\leq n} \subseteq T_n$.*

Proof. If $\sigma \in \hat{T}_n \cap J_n^{\leq n}$, then $\sigma \in R_\beta$ for some β along the true path. When that β is first visited, the string σ is enumerated into T_n. □

Claim 4.5. *Fix a Υ_n-strategy α along the true path. If $\sigma^\frown\langle j\rangle, \sigma^\frown\langle k\rangle \in \hat{T}_n \cap J_n^{\leq n-1}$, then*

$$[\sigma^\frown\langle j\rangle] - [\sigma^\frown\langle j\rangle^\frown\langle k\rangle] \cong [\sigma^\frown\langle k\rangle] - [\sigma^\frown\langle k\rangle^\frown\langle j\rangle]$$

via the map $\sigma^\frown\langle j\rangle^\frown\rho \mapsto \sigma^\frown\langle k\rangle^\frown\rho$, where the indicated isomorphisms are isomorphisms of the tree T_n.

Proof. Let β be a $\Xi_{n,*}$-strategy along the true path such that $\sigma^\frown\langle j\rangle, \sigma^\frown\langle k\rangle \in J_\beta^{\leq n-1}$. Suppose $\sigma^\frown\langle j\rangle^\frown\rho \in [\sigma^\frown\langle j\rangle] - [\sigma^\frown\langle j\rangle^\frown\langle k\rangle]$. Let s be the stage at which $\sigma^\frown\langle j\rangle^\frown\rho$ enters T_n. Let $t > s$ be a stage at which β is visited. By action (2) of β, $\sigma^\frown\langle k\rangle^\frown\rho$ will enter T_n at stage t if it has not already done so. Thus $\sigma^\frown\langle k\rangle^\frown\rho \in [\sigma^\frown\langle k\rangle]$. Since $\sigma^\frown\langle j\rangle^\frown\rho \in T_n$, $\rho(0) \neq j$, and so $\sigma^\frown\langle k\rangle^\frown\rho \in [\sigma^\frown\langle k\rangle] - [\sigma^\frown\langle k\rangle^\frown\langle j\rangle]$.

A symmetric argument applies to $\sigma^\frown\langle k\rangle^\frown\rho \in [\sigma^\frown\langle k\rangle] - [\sigma^\frown\langle k\rangle^\frown\langle j\rangle]$. □

Claim 4.6. *Fix a Υ_n-strategy α. No string containing j can be enumerated into T_n until the $\Xi_{n,j}$-strategy has been visited for the first time.*

Proof. Immediate by induction on the order in which the $\Xi_{n,j}$-strategies are visited. □

Claim 4.7. *Suppose α is a $\Xi_{n,j}$-strategy. If α enumerates some $\sigma^\frown\langle k\rangle^\frown\rho$ into T_n by action (2), then $\rho \notin J_\beta^{\leq n}$ for any $\Xi_{n,j'}$-strategy β comparable to α.*

Proof. Suppose otherwise. Then no string containing ρ as a substring can be enumerated into T_n before the least such β is first visited. But then $\sigma^\frown\langle k\rangle^\frown\rho$ will be enumerated into T_n by action (1) by the first time $\sup\{\alpha, \beta\}$ is visited. □

Definition 4.8. For $\sigma \in T_n$, the *tail of* σ (denoted tail(σ)) is the maximal final segment ρ of σ with $\rho \in R_\beta$ for some β.

The *origin of* σ (denoted origin(σ)) is the (unique) $\Xi_{n,j}$-strategy β with tail$(\sigma) \in R_\beta$.

Note 4.9. If σ was enumerated into T_n by action (1) of strategy β, then tail$(\sigma) = \sigma$ and origin$(\sigma) = \beta$. By Claim 4.10 below, β is also the origin of all "copies" of σ inserted via action (2).

Claim 4.10. *Suppose α is a Ξ-strategy. If α enumerates some σ into T_n by action (2), copying some τ already in T_n (which has entered T_n via either action (1) or (2) of another strategy), then origin(σ) = origin(τ).*

Proof. Let origin(τ) be a $\Xi_{n,j}$-strategy. Let m be greatest such that $\tau(m) = j$ (necessarily in the tail of τ). Then α creates σ from τ by replacing some single character $\tau(p)$ with some other character l. By assumption, $\tau \restriction (p+1) \in R_\beta$ for some $\beta \subseteq \alpha$.

If $p \geq m$, then origin$(\tau) \subseteq \beta$, and so $\tau \in R_\beta$. Thus $\sigma \in R_\gamma$ for some $\gamma \subseteq \alpha$, and so σ would have been enumerated into T_n by action (1) when γ was first visited, contrary to hypothesis.

Thus $p < m$. Let ρ be such that $\tau = (\tau \restriction m)^\frown\rho$. So $\rho(0) = j$ and origin(ρ) = origin(τ). Then ρ is a (not necessarily proper) final segment of tail(τ). ρ is also a final segment of σ. From the maximality of tail(σ), ρ is a final segment of tail(σ), and so origin(τ) = origin$(\rho) \subseteq$ origin(σ).

If tail(σ) is a final segment of τ, then by the maximality of tail(τ), tail(σ) is a final segment of tail(τ). So origin$(\sigma) \subseteq$ origin(τ), as desired.

Else, $|\mathrm{tail}(\sigma)| \geq |\sigma| - p$ and $|\mathrm{tail}(\tau)| \geq |\tau| - p - 1$. If $\mathrm{origin}(\tau) \subsetneq \mathrm{origin}(\sigma)$, then fix j' such that $\mathrm{origin}(\sigma)$ is a $\Xi_{n,j'}$-strategy. Then j' occurs somewhere in $\mathrm{tail}(\sigma)$ but not in $\mathrm{tail}(\tau)$. Hence j' occurs somewhere in $\sigma \restriction (p+1)$. Thus $\sigma \restriction (p+1) \in R_{\beta'}$ for some β' extending $\mathrm{origin}(\sigma)$. So $\sigma \in R_{\beta'}$. But by assumption $\beta' \subseteq \alpha$. Thus σ would have been enumerated into T_n by action (1) when $\mathrm{origin}(\sigma)$ was first visited, contrary to hypothesis.

Hence $\mathrm{origin}(\tau) = \mathrm{origin}(\sigma)$. □

Claim 4.11. *Suppose α is a Ξ-strategy. If α enumerates some σ into T_n by action (2), then tail$(\sigma) \neq \sigma$.*

Proof. If not, then σ would have already been enumerated into T_n by action (1) of $\mathrm{origin}(\sigma)$. □

For a string $\sigma_0 \in T_n$, σ_0 may have been enumerated by some strategy α_0 performing action (2) and copying a string σ_1 already in T_n. σ_1 in turn may have been enumerated by some strategy α_1 performing action (2) and copying σ_2. In this fashion, we may generate a sequence $\sigma_0, \sigma_1, \dots, \sigma_m, \sigma_{m+1}$, with each string being enumerated because a strategy copied the next. Of course, this process must terminate in some σ_{m+1} which was enumerated by action (1) of some strategy α_{m+1}. It would be helpful to track the strategy α_m which performed the first copy in the sequence, enumerating σ_m by copying σ_{m+1}. In particular, we would like to be able to determine this strategy simply by examining σ_0, without considering the history of the construction. This is a little too much to hope for, but by examining σ_0, we can determine a strategy $\beta \subseteq \alpha_m$, which we call the pillager of σ_0 and which is incomparable with $\mathrm{origin}(\sigma_0)$, which turns out to be good enough for our purposes.

Definition 4.12. If $\sigma \neq \mathrm{tail}(\sigma)$, then the *pillager of* σ (denoted as $\mathrm{pillager}(\sigma)$) is the unique $\Xi_{n,j}$-strategy where j is such that σ can be written as $\sigma = \sigma'^\frown\langle j\rangle^\frown\mathrm{tail}(\sigma)$ for some σ'.

Claim 4.13. *If $\sigma \neq$ tail(σ), then pillager(τ) and origin(τ) are incomparable.*

Proof. By the maximality of the tail. □

Claim 4.14. *Suppose α is a Ξ-strategy. If α enumerates some σ into T_n by action (2), copying some τ already in T_n, and $\tau =$ tail(τ) or pillager$(\sigma) \neq$ pillager(τ), then pillager$(\sigma) \subseteq \alpha$.*

Proof. If $\tau = \mathrm{tail}(\tau)$, then the final position before the tail of σ (the position which determines $\mathrm{pillager}(\sigma)$) is the position changed by the action of α. So $\mathrm{pillager}(\sigma) \subseteq \alpha$.

If $\mathrm{pillager}(\sigma) \neq \mathrm{pillager}(\tau)$ and $|\mathrm{tail}(\sigma)| \leq |\mathrm{tail}(\tau)|$, then by the maximality of $\mathrm{tail}(\sigma)$, the position before the tail of σ is the position changed by the action of α. So $\mathrm{pillager}(\sigma) \subseteq \alpha$.

If $\mathrm{pillager}(\sigma) \neq \mathrm{pillager}(\tau)$ and $|\mathrm{tail}(\sigma)| > |\mathrm{tail}(\tau)|$, then by the maximality of $\mathrm{tail}(\tau)$, the position before the tail of τ is the position changed by the action of α. So the position before the tail of σ is earlier. So by the assumption of action (2), the character before the tail of σ is in J_α. Thus $\mathrm{pillager}(\sigma) \subseteq \alpha$. □

Claim 4.15. *If $\sigma \in T_n$ and $\sigma \neq$ tail(σ), then let t be the stage at which σ was enumerated into T_n. Then there are stages $s_0 < s_1 \leq t$ such that origin(σ) was visited at stage s_0 and pillager(σ) was visited at stage s_1.*

Proof. Proof by induction on t. Some strategy α created σ by copying some string τ through action (2) at stage t.

If $\tau = \text{tail}(\tau)$, then $\text{origin}(\sigma) = \text{origin}(\tau)$ was visited at some stage s_0 when τ was enumerated into T_n. By Claim 4.14, $\text{pillager}(\sigma)$ was visited at stage t. Thus we can take $s_1 = t$.

If $\tau \neq \text{tail}(\tau)$ and $\text{pillager}(\sigma) \neq \text{pillager}(\tau)$, then a simple induction using Claim 4.10 shows that $\text{origin}(\sigma) = \text{origin}(\tau)$ was visited at some stage s_0 before the stage when τ was enumerated into T_n. By Claim 4.14, $\text{pillager}(\sigma)$ was visited at stage t. Thus we can take $s_1 = t$.

If $\tau \neq \text{tail}(\tau)$ and $\text{pillager}(\sigma) = \text{pillager}(\tau)$, then the result follows immediately from the inductive hypothesis applied to τ. □

Claim 4.16. *Suppose α is a Ξ-strategy. If α enumerates some σ into T_n by action (2), copying some τ already in T_n, and $\tau \neq \textit{tail}(\tau)$, and $\textit{origin}(\tau)$ has been visited between the enumerations of τ and σ into T_n, then $\textit{pillager}(\sigma) = \textit{pillager}(\tau)$.*

Proof. Let t be the stage at which τ was enumerated into T_n, and let $s_0 < s_1 \leq t$ be the stages from Claim 4.15 applied to τ. By assumption, there is some stage $s_2 > t$ at which $\text{origin}(\tau)$ is visited. Thus $\text{origin}(\tau)$ was visited both before and after $\text{pillager}(\tau)$, and by Claim 4.13, the two are incomparable. Thus $\text{origin}(\tau)$ must be to the left of $\text{pillager}(\tau)$ in the priority tree, and so $\text{pillager}(\tau)$ can never again be visited after stage s_2. In particular, $\text{pillager}(\tau)$ cannot be visited at the stage at which σ is enumerated, and so $\text{pillager}(\tau) \not\subseteq \alpha$. Thus the final character before the tail of τ (the character which determines $\text{pillager}(\tau)$) is not in J_α, and so the single character α replaces must occur before that one. Thus $\text{tail}(\sigma) = \text{tail}(\tau)$ and the final character in σ before $\text{tail}(\sigma)$ is the same as the character in τ. So $\text{pillager}(\sigma) = \text{pillager}(\tau)$. □

Claim 4.17. *Suppose α is a Ξ-strategy. If α enumerates some σ into T_n by action (2), copying some τ already in T_n, then $\sigma(|\sigma| - 1) = \tau(|\tau| - 1)$.*

Proof. Immediate by construction since else σ would have been enumerated into T_n by action (1). □

Claim 4.18. *If $\sigma \in T_n$, then $\sigma(|\sigma| - 1) \in J_{origin(\sigma)}$.*

Proof. Immediate by Claim 4.17 and induction on the number of times action (2) was performed before σ was added to T_n, using Claim 4.10. □

Definition 4.19. If α is a $\Xi_{n,j}$-strategy and $k \in J_\alpha$ with $k \neq j$, the *canonical label of type $\langle k, \alpha \rangle$* is the temporary label attached to $z_{\langle j,k \rangle}$. The *canonical label of type $\langle j, \alpha \rangle$* is the temporary label attached to $z_{\langle j \rangle}$.

Claim 4.20. *Suppose $\sigma \in T_n$ at stage s.*

If $\sigma = \textit{tail}(\sigma)$, then the temporary label attached to σ at stage s is identical to the canonical label of type $\langle \sigma(|\sigma| - 1), \textit{origin}(\sigma) \rangle$ at stage s.

If $\sigma \neq \textit{tail}(\sigma)$, and $t \leq s$ is the last stage at which $\textit{pillager}(\sigma)$ is visited, then the temporary label attached to σ at stage s is identical to the canonical label of type $\langle \sigma(|\sigma| - 1), \textit{origin}(\sigma) \rangle$ at stage t.

Proof. We perform induction on the number of applications of action (2).

If $\sigma = \text{tail}(\sigma)$, then σ was enumerated by action (1) of $\text{origin}(\sigma)$. Thus its label was created at the same time as the canonical label of its type, and both labels grow by one step precisely at the stages when origin(σ) is visited. This proves the first part of the claim.

If $\sigma \neq$ tail(σ), then σ was enumerated by action (2) of some strategy α copying some τ already in T_n. Let $s' \leq s$ be the stage at which σ was enumerated into T_n.

If $\tau =$ tail(τ), then by the first part of the claim, the temporary label attached to τ (and thus the temporary label attached to σ) is identical to the appropriate canonical label at stage s'. Furthermore, s' is a stage at which pillager(σ) is visited by Claim 4.14. Since, by Claim 4.13, pillager(σ) and origin(σ) are incomparable, origin(σ) cannot be visited between stages s' and t. Thus the canonical label cannot grow at all by stage t (and by construction, the temporary label attached to σ never grows). So the temporary label attached to σ is identical to the appropriate canonical label at stage t.

If $\tau \neq$ tail(τ), let $t' \leq s'$ be the greatest stage such that pillager(τ) was visited at stage t'.

If pillager(σ) = pillager(τ) then, by construction, the temporary label attached to σ is identical to the temporary label attached to τ, and by the inductive hypothesis, this is equal to the appropriate canonical label at stage t'. Since, by Claim 4.13, pillager(σ) and origin(σ) are incomparable, origin(σ) cannot be visited between stages t' and t. Thus the canonical label cannot grow at all by stage t (and by construction the temporary label attached to σ never grows). So the temporary label attached to σ is identical to the appropriate canonical label at stage t.

If pillager(σ) $\neq$ pillager(τ), then, by Claim 4.16, origin(τ) was not visited between stages t' and s'. By the inductive hypothesis, the temporary label attached to σ is identical to the temporary label attached to τ, which is identical to the appropriate canonical label at stage t'. But since origin(τ) is not visited, this is identical to the canonical label at stage s'. Furthermore, pillager(σ) is visited at stage s'. So origin(σ) cannot be visited between stages s' and t, so the canonical label cannot grow. So the temporary label attached to σ is identical to the canonical label at stage t. This proves the second part of the claim. ☐

Claim 4.21. *Fix a Υ_n-strategy α along the true path. The isomorphism constructed in Claim 4.5 extends to isomorphisms of the retinues.*

Proof. It suffices to show that the temporary labels attached to $\sigma^\frown\langle j\rangle^\frown\rho$ and $\sigma^\frown\langle k\rangle^\frown\rho$ are identical where $|\rho| > 0$. Without loss of generality, the $\Xi_{n,j}$-strategy is an extension of the $\Xi_{n,k}$-strategy. Call these strategies α_j and α_k, respectively.

If $\rho \in J_n^{<\omega}$, then $\sigma^\frown\langle j\rangle^\frown\rho \in R_{\beta_j}$ and $\sigma^\frown\langle k\rangle^\frown\rho \in R_{\beta_k}$ for some strategies β_j, β_k along the true path. So the temporary labels attached to each will be grown infinitely often, and thus they will both consist of a single loop of every size in S.

If $|\text{tail}(\sigma^\frown\langle j\rangle^\frown\rho)| < |\rho|$, then $\text{tail}(\sigma^\frown\langle k\rangle^\frown\rho) = \text{tail}(\sigma^\frown\langle j\rangle^\frown\rho)$, and so we have $\text{origin}(\sigma^\frown\langle j\rangle^\frown\rho) = \text{origin}(\sigma^\frown\langle k\rangle^\frown\rho)$ and $\text{pillager}(\sigma^\frown\langle j\rangle^\frown\rho) = \text{pillager}(\sigma^\frown\langle k\rangle^\frown\rho)$. By Claim 4.20, the temporary labels are identical.

If $\text{tail}(\sigma^\frown\langle k\rangle^\frown\rho) = \sigma^\frown\langle k\rangle^\frown\rho$ and $\rho \notin J_n^{<\omega}$, then the character which determines $\text{origin}(\sigma^\frown\langle k\rangle^\frown\rho)$ must occur somewhere in ρ. Therefore, $\text{origin}(\sigma^\frown\langle k\rangle^\frown\rho) = \text{origin}(\sigma^\frown\langle j\rangle^\frown\rho)$. If $\text{tail}(\sigma^\frown\langle j\rangle^\frown\rho) = \sigma^\frown\langle j\rangle^\frown\rho$, then by Claim 4.20, both temporary labels are identical to the canonical label at all stages. If $\text{tail}(\sigma^\frown\langle j\rangle^\frown\rho) \neq \sigma^\frown\langle j\rangle^\frown\rho$, then $\alpha_j = \text{pillager}(\sigma^\frown\langle j\rangle^\frown\rho)$, so by Claim 4.20 the temporary labels are identical to the canonical label at any stage at which α_j is visited.

If $|\text{tail}(\sigma^\frown\langle j\rangle^\frown\rho)| \geq |\rho|$ and $\text{tail}(\sigma^\frown\langle k\rangle^\frown\rho) \neq \sigma^\frown\langle k\rangle^\frown\rho$, then $|\text{tail}(\sigma^\frown\langle k\rangle^\frown\rho)| \geq |\rho|$. But then both $\text{pillager}(\sigma^\frown\langle j\rangle^\frown\rho)$ and $\text{pillager}(\sigma^\frown\langle k\rangle^\frown\rho)$ are on the true path.

Since origin and tail are incomparable, origin($\sigma^\frown\langle j\rangle^\frown\rho$) is not on the true path, and thus the character defining the origin must occur somewhere in ρ. Therefore, origin($\sigma^\frown\langle k\rangle^\frown\rho$) = origin($\sigma^\frown\langle j\rangle^\frown\rho$). Then at any stage when the longer of the two pillagers is visited, both temporary labels are identical to the canonical label by Claim 4.20. So in the limit, the two labels are the same. □

Claim 4.22. *Fix a Υ_n-strategy α along the true path. The tree T_n satisfies properties* (I)-(IV).

Proof. By Claims 4.3, 4.4, 4.5, and 4.21. □

Next, we show that the types behave as previously claimed.

Claim 4.23. *Fix a Υ_n-strategy α along the true path.*

If $\sigma, \tau \in T_n$ of the same length differ in a single coordinate $k < n-1$, and $\sigma \restriction (k+1), \tau \restriction (k+1) \in J_n^{k+1}$, then the retinue of σ is isomorphic to the retinue of τ.

Proof. By Property (IV) of T_n (verified in Claim 4.22), there is an isomorphism

$$[\sigma \restriction (k+1)] - [\sigma \restriction (k+1)^\frown\langle\tau(k)\rangle] \to [\tau \restriction (k+1)] - [\tau \restriction (k+1)^\frown\langle\sigma(k)\rangle],$$

and this isomorphism extends to the retinues.

If σ and τ have length $k+1$, then they are in the domain and range of this isomorphism, respectively. Otherwise, since $\sigma(k+1) = \tau(k+1)$, it must be the case that $\sigma(k+1) \neq \tau(k)$ and $\tau(k+1) \neq \sigma(k)$. Thus σ and τ are again in the domain and range, respectively. □

Claim 4.24. *Fix a Υ_n-strategy α along the true path.*

Let $u_1, \ldots, u_{m'}, v_1, \ldots, v_m \in \mathcal{A}$ and $\sigma_1, \ldots, \sigma_m \in T_n$ where v_ℓ is in the retinue of σ_ℓ and $u_\ell \notin$ retinue(T_n) for all ℓ. Suppose that $k < n-1$ and $j, j' \in J_n$ such that if $|\sigma_\ell| \geq k+1$ and $\sigma_\ell \restriction (k+1) \in J_n^{k+1}$, then $j' \notin \{\sigma_\ell(k-1), \sigma_\ell(k), \sigma_\ell(k+1)\}$ (ignoring those $\sigma_\ell(k-1)$ or $\sigma_\ell(k+1)$ which are not defined).

Let τ_ℓ be obtained from σ_ℓ by replacing any j in the kth place by j' unless $\sigma_\ell \restriction (k+1) \notin J_n^{k+1}$. More precisely, let $\tau_1, \ldots, \tau_m \in T_n$ be defined as follows:

$$\tau_\ell(i) := \begin{cases} \sigma_\ell(i) & \text{if } i \neq k, \\ \sigma_\ell(i) & \text{if } \sigma_\ell(i) \neq j, \\ \sigma_\ell(i) & \text{if } \sigma_\ell \restriction (i+1) \notin J_n^{i+1}, \\ j' & \text{otherwise.} \end{cases}$$

Using Claim 4.23, let w_ℓ be the natural image of v_ℓ in the retinue of τ_ℓ.

Then Σ^c_{n-k-2}-type($\overline{u}, \overline{v}$) = Σ^c_{n-k-2}-type($\overline{u}, \overline{w}$).

Proof. We proceed by induction on the complexity of types.

For Σ_0-types, this is trivial.

Let $\varphi(\overline{y}, \overline{z}) = \exists \overline{x}\ \psi(\overline{y}, \overline{z}, \overline{x})$ with ψ a Π^c_p-formula with $p < n-k-2$, and suppose $\mathcal{A} \models \varphi(\overline{u}, \overline{v})$. Then let $\overline{u}', \overline{v}'$ be such that $\mathcal{A} \models \psi(\overline{u}, \overline{v}, \overline{u}'\overline{v}')$, with $\overline{u}' \notin U_i$ and $\overline{v}' \in U_i$, and let $\overline{\sigma}'$ be such that the various v' are in the retinue of the various σ'. By the inductive hypothesis, if $\sigma'_\ell \restriction (k+1) \in J_n^{k+1}$, we may assume that $j' \notin \{\sigma_\ell(k-1), \sigma_\ell(k), \sigma_\ell(k+1)\}$ (by employing the inductive hypothesis three times with three large values from J_n). (Since $n-k-2 > 0$, $k+1 < n-1$.)

Then let $\overline{\tau}'$ correspond to $\overline{\sigma}'$ in the same fashion as $\overline{\tau}$ corresponds to $\overline{\sigma}$ (if $\sigma'_\ell \restriction (k+1) \in J_n^{k+1}$, all j at coordinate k are replaced with j'). Let $\overline{w}'$ be the natural images of the $\overline{v}'$ in the retinues of the $\overline{\tau}'$. Then by the inductive hypothesis, $\mathcal{A} \models \psi(\overline{u}, \overline{w}, \overline{u}'\overline{w}')$. Thus $\mathcal{A} \models \varphi(\overline{u}, \overline{w})$.

The proof of the other direction is symmetric. $\square$

Now we show that $\mathcal{A}$ is not relatively arithmetically categorical.

Claim 4.25. *For distinct $j, j' \in \omega$, if $\sigma^\frown\langle j\rangle, \sigma^\frown\langle j'\rangle \in T_n$, then there is no embedding $[\sigma^\frown\langle j\rangle] \hookrightarrow [\sigma^\frown\langle j'\rangle]$ that extends to an embedding of the retinues.*

Proof. Towards a contradiction, let f be such an embedding. Let $k < n$ be greatest such that for some τ, $\tau(k) \neq f(\tau)(k)$. Fix such a string τ, where we assume without loss of generality that $|\tau| = k+1$. If $k = n-1$ (i.e., if $|\tau| = n$), then z_τ has a leaf loop of size $w^L_{\tau(k)}$ whereas $z_{f(\tau)}$ has a leaf loop of size $w^L_{f(\tau)(k)}$. As these sizes are distinct, the embedding f cannot extend to an embedding of the retinues. If $k < n-1$ (i.e., if $|\tau| < n$), then by maximality of k, we must have $f(\tau^\frown\langle f(\tau)(k)\rangle) = f(\tau)^\frown\langle f(\tau)(k)\rangle$. But $f(\tau)^\frown\langle f(\tau)(k)\rangle \notin T_n$ since it repeats a character in the k^{th} and $(k+1)^{st}$ positions. $\square$

Claim 4.26. *The structure $\mathcal{A}$ has no non-trivial self-embeddings.*

Proof. Let $f : \mathcal{A} \to \mathcal{A}$ be an embedding. Every point in $\mathcal{A}$ which is a z_σ for some σ has out-degree strictly greater than 1 (from its temporary label and height label, if nothing else), while all other points all have out-degree exactly 1. Thus f must map every z_σ to some z_τ. Moreover, by examining the height labels, it must be the case that σ and τ belong to the same tree T_n. Hence f induces embeddings $f : T_n \to T_n$ for each $n \in \omega$.

By Claim 4.25, it follows that f must be the identity on T_n. As the retinue of any $\sigma \in T_n$ has no non-trivial self-embeddings (loops are always of distinct sizes), this implies f is the identity. $\square$

Claim 4.27. *For every n, the structure $\mathcal{A}$ does not have a c.e. Scott family of Σ^c_n-formulas. Thus, the structure $\mathcal{A}$ is not relatively arithmetically categorical.*

Proof. Fix n. Suppose X were a family of Σ^c_n-formulas with parameters $\overline{c}$. Fix $n' \geq n+2$ such that the $\Upsilon_{n'}$-strategy is along the true path and the retinue of $T_{n'}$ is disjoint from $\overline{c}$. Choose $j \in J_{n'}$ with $j \neq 0$. Then by Claim 4.24, we have $\Sigma^c_{n'}$-type$(\overline{c}, z_{\langle 0\rangle}) = \Sigma^c_{n'}$-type$(\overline{c}, z_{\langle j\rangle})$. Thus for any formula $\varphi \in X$, $\mathcal{A} \models \varphi(\overline{c}, z_{\langle 0\rangle}) \leftrightarrow \varphi(\overline{c}, z_{\langle j\rangle})$. But by Claim 4.26, $z_{\langle 0\rangle}$ and $z_{\langle j\rangle}$ are not in the same orbit. Thus X is not a Scott family.

It follows from Proposition 1.7 that $\mathcal{A}$ is not relatively arithmetically categorical. $\square$

Finally, we show that $\mathcal{A}$ is computably categorical.

Claim 4.28. *If $\sigma \in T_n$ was enumerated at stage s, tail$(\sigma) \neq \sigma$, α is a $\Xi_{n,j}$-strategy and j appears in σ outside tail(σ), let $t_0 < s$ be the first stage at which origin(σ) was visited, $t_1 > t_0$ be least such that pillager(σ) was visited at stage t_1 (t_1 exists by Claim 4.15), and $t_2 \leq s$ be greatest such that α was visited at stage t_2. Then $t_1 \leq t_2$.*

Proof. By induction on the number of applications of action (2).

If some β enumerates σ into T_n at stage s by copying some τ, let $\tau(m)$ be the character that was changed to create σ.

If $|\text{tail}(\sigma)|+m+1 \geq |\sigma|$ (this includes the base case $\tau = \text{tail}(\tau)$ since then tail(σ) begins right after $\sigma(m)$), then j and the character which determines the pillager of σ occurs within $\sigma \upharpoonright (m+1)$. Since $\sigma \upharpoonright (m+1) \in R_\beta$, $\alpha \subseteq \beta$ and $\text{pillager}(\sigma) \subseteq \beta$. Thus $t_1 \leq s = t_2$.

Otherwise, by the maximality of tails, $\text{tail}(\sigma) = \text{tail}(\tau)$, and the last characters before the tails of σ and τ are the same. So we have $\text{pillager}(\sigma) = \text{pillager}(\tau)$ and $\text{origin}(\sigma) = \text{origin}(\tau)$. If j occurs in τ outside of $\text{tail}(\tau)$ (i.e., in particular, j is not the character replaced), we have by the inductive hypothesis (applied to τ in place of σ) that $t_1 \leq t_2$. Otherwise, $j = \sigma(m)$. Then $j \in J_\beta$, and so $\alpha \subseteq \beta$. Thus $t_2 = s$, while $t_1 \leq s$ by the inductive hypothesis (applied to τ in place of σ and $\tau(0)$ in place of j). □

Claim 4.29. *If $\sigma^\frown\langle j\rangle \in T_n$ at stage s, let m be largest such that there is a loop of size m in the temporary label attached to $z_{\sigma^\frown\langle j\rangle}$ at stage s. Then $z_{\sigma^\frown\langle j\rangle}$ is the unique child of z_σ with a loop of size m in its temporary label at stage s. (Or, for $\sigma = \langle\rangle$, $z_{\langle j\rangle}$ is the unique height-1 element with a loop of size m.)*

Proof. By Claim 4.20, there was a stage at which m was the size of the largest loop in the canonical label of type $\langle j, \text{origin}(\sigma^\frown\langle j\rangle)\rangle$. By the construction of temporary labels, $m \in W^{\langle j, \text{origin}(\sigma^\frown\langle j\rangle)\rangle}$. Also by the construction of temporary labels, a temporary label of any type other than $\langle j, \text{origin}(\sigma^\frown\langle j\rangle)\rangle$ can only gain a loop of size m after some temporary label of type $\langle j, \text{origin}(\sigma^\frown\langle j\rangle)\rangle$ enumerates m into S. This can only happen after that temporary label has added a loop of some size bigger than m. By Claim 4.20, when this happens, the canonical label will have a loop of a size larger than m.

If $\sigma^\frown\langle j\rangle = \text{tail}(\sigma^\frown\langle j\rangle)$, then we have by Claim 4.20 that the temporary label attached to $z_{\sigma^\frown\langle j\rangle}$ will be identical to the canonical label of type $\langle j, \text{origin}(\sigma^\frown\langle j\rangle)\rangle$, and so m is the size of the largest loop in the canonical label at stage s. By the above, no label of a type other than $\langle j, \text{origin}(\sigma^\frown\langle j\rangle)\rangle$ has a loop of size m at stage s. Every sibling of $z_{\sigma^\frown\langle j\rangle}$ has a label of a type other than $\langle j, \text{origin}(\sigma^\frown\langle j\rangle)\rangle$.

Otherwise, consider any other $\sigma^\frown\langle k\rangle \in T_n$ at stage s. There are three possibilities.

If $|\text{tail}(\sigma^\frown\langle j\rangle)| = |\text{tail}(\sigma^\frown\langle k\rangle)|$, then note that $\text{pillager}(\sigma^\frown\langle j\rangle) = \text{pillager}(\sigma^\frown\langle k\rangle)$. Let t be the last stage at which this common pillager was visited. By Claim 4.20, m is the largest loop attached to the canonical label of type $\langle j, \text{origin}(\sigma^\frown\langle j\rangle)\rangle$ at stage t, and by the construction of temporary labels, the canonical label of type $\langle k, \text{origin}(\sigma^\frown\langle k\rangle)\rangle$ does not possess a loop of size m at stage t. By Claim 4.20 again, the temporary label attached to $z_{\sigma^\frown\langle k\rangle}$ does not have a loop of size m at stage s.

If $|\text{tail}(\sigma^\frown\langle j\rangle)| < |\text{tail}(\sigma^\frown\langle k\rangle)|$, then since $|\text{tail}(\sigma^\frown\langle j\rangle)| > 0$, the final j is not the character which determines $\text{pillager}(\sigma^\frown\langle j\rangle)$. Thus the character which determines this pillager is contained in $\text{tail}(\sigma^\frown\langle k\rangle)$, and so $\text{pillager}(\sigma^\frown\langle j\rangle) \subseteq \text{origin}(\sigma^\frown\langle k\rangle)$. If the temporary label attached to $z_{\sigma^\frown\langle k\rangle}$ has a loop of size m, then the canonical label of type $\langle k, \text{origin}(\sigma^\frown\langle k\rangle)\rangle$ has a loop of size m. Let t be the stage at which this loop was added to the canonical label. Then at stage t, $\text{origin}(\sigma^\frown\langle k\rangle)$ was visited, and thus $\text{pillager}(\sigma^\frown\langle j\rangle)$ was visited. However, by the construction of temporary labels, if a label of type other than $\langle j, \text{origin}(\sigma^\frown\langle j\rangle)\rangle$ has a loop of size m at stage t, then the canonical label of type $\langle j, \text{origin}(\sigma^\frown\langle j\rangle)\rangle$ must have a loop of size greater than m at stage t. By Claim 4.20, the temporary label attached to $z_{\sigma^\frown\langle j\rangle}$ would then have a loop of size greater than m, contradicting our choice of m.

If $|\text{tail}(\sigma^\frown\langle j\rangle)| > |\text{tail}(\sigma^\frown\langle k\rangle)|$, then let $\sigma^\frown\langle j\rangle = \sigma'^\frown\langle p\rangle^\frown\text{tail}(\sigma^\frown\langle j\rangle)$. Then pillager$(\sigma^\frown\langle j\rangle)$ is a $\Xi_{n,p}$-strategy, and p appears in $\sigma^\frown\langle k\rangle$ outside tail$(\sigma^\frown\langle k\rangle)$. If the temporary label attached to $z_{\sigma^\frown\langle k\rangle}$ has a loop of size m, then at some stage t_0, the canonical label of type $\langle k, \text{origin}(\sigma^\frown\langle k\rangle)\rangle$ gained a loop of this size. This is a stage at which origin$(\sigma^\frown\langle k\rangle)$ is visited. Furthermore, by Claim 4.20, since this loop of size m occurs in the temporary label attached to $z_{\sigma^\frown\langle k\rangle}$, pillager$(\sigma^\frown\langle k\rangle)$ must be visited at some stage $t_1 > t_0$. Then by Claim 4.28, at some stage t_2 with $t_1 \leq t_2 \leq s$, pillager$(\sigma^\frown\langle j\rangle)$ is visited. However, at stage t_0 (and thus at stage t_2), since the canonical label of type $\langle k, \text{origin}(\sigma^\frown\langle k\rangle)\rangle$ has a loop of size m, the canonical label of type $\langle j, \text{origin}(\sigma^\frown\langle j\rangle)\rangle$ will have a loop of size greater than m, and thus by Claim 4.20, at stage s the loop attached to $z_{\sigma^\frown\langle j\rangle}$ will have a loop of size greater than m, contradicting our choice of m. □

Definition 4.30. If $\mathcal{A} \cong \mathcal{B}_\ell$, let f be the unique isomorphism $f : \mathcal{A} \to \mathcal{B}_\ell$. Let α be the Φ_ℓ-strategy along the true path.

We call a non-provisional tag z_σ *correctly placed* if the tagged element is $f(z_\sigma)$.

We call a provisional tag $z_{m,\tau}$ *correctly placed* if there is some σ with $|\sigma| = m$ and $\sigma \in R_\beta$ for some $\beta \subset \alpha$ such that the tagged element is $f(z_{\sigma^\frown\tau})$.

Claim 4.31. *If a Φ_ℓ-strategy α is along the true path and $\mathcal{B}_\ell \cong \mathcal{A}$, then every non-provisional tag is correctly placed.*

Proof. We proceed by induction on the length of the σ being tagged.

If the tag is never placed, then z_σ has a loop of some fixed size m in its temporary label which does not occur in $f(z_\sigma)$, contradicting $\mathcal{B}_\ell \cong \mathcal{A}$.

If the tag is incorrectly placed, then by Claim 4.25, there must be some non-provisional tag z_ρ with $\rho \subseteq \sigma$ that either α will fail to place, or eventually the retinues will cease appearing isomorphic. Either way, α will have true outcome k for some finite k. Thus the temporary label will never grow, and z_σ will be the only child of z_{σ^-} with a loop of size m for some fixed m.

By construction, the element tagged as z_σ has a loop of size m, and by the inductive hypothesis, the tag z_{σ^-} is correctly placed. Since $\mathcal{B}_\ell \cong \mathcal{A}$ and by Claim 4.29, the tagged element is the only one of its siblings to have a loop of size m.

Thus any embedding of $\mathcal{A}$ into $\mathcal{B}_\ell$ must send z_σ to the tagged element, so the element tagged as z_σ must be the correct element. □

Claim 4.32. *If a Φ_ℓ-strategy α is along the true path, and $z_{m,\tau}$ is a provisional tag that α seeks to place, then for any $\sigma_0, \sigma_1 \in T_n$ with $|\sigma_0| = |\sigma_1| = m$ and σ_0, σ_1 each in R_β for some $\beta \subset \alpha$, at any stage when α is visited, the temporary labels attached to $z_{\sigma_0{}^\frown\tau}$ and $z_{\sigma_1{}^\frown\tau}$ are identical.*

Proof. By construction of the temporary tags, $\tau(0) \in R_\gamma$ for some γ incomparable to $\alpha^\frown\langle k\rangle$ (where k is the current finite outcome of α). Thus $\beta \subseteq \gamma$, or β and γ are incomparable; so origin(τ) = origin$(\sigma_0{}^\frown\tau)$ = origin$(\sigma_1{}^\frown\tau)$.

If $|\text{tail}(\tau)| < |\tau|$, then we have tail$(\tau)$ = tail$(\sigma_0{}^\frown\tau)$ = tail$(\sigma_1{}^\frown\tau)$ and pillager(τ) = pillager$(\sigma_0{}^\frown\tau)$ = pillager$(\sigma_1{}^\frown\tau)$. So the temporary labels are always identical by Claim 4.20.

If tail$(\tau) = \tau$, then the characters determining pillager$(\sigma_0{}^\frown\tau)$ and pillager$(\sigma_1{}^\frown\tau)$ are in σ_0 and σ_1, respectively. Thus both of these pillagers are initial segments of α, and so are visited at any stage at which α is visited. At such a stage, by Claim 4.20, the temporary labels are identical. □

Claim 4.33. *If a Φ_ℓ-strategy α is along the true path and $\mathcal{B}_\ell \cong \mathcal{A}$, then any provisional tag not correctly placed is eventually moved.*

Proof. Suppose otherwise. Let $b_0 \in \mathcal{B}_\ell$ have least height such that b_0 gains a provisional tag which is not correctly placed and never moved.

If the tag in question is $z_{m,\tau}$, then $b_0 = f(z_{\sigma^\frown\tau'})$ for some σ and τ' with $|\sigma| = m$, $|\tau'| = |\tau|$, $\sigma \in R_\beta$ for some $\beta \subset \alpha$, and such that τ and τ' differ only in the final character. The placement of the $z_{m,\rho}$ tags with $\tau \subset \rho$ gives a partial embedding

$$[\sigma^\frown\tau] \hookrightarrow [\sigma^\frown\tau'].$$

By Claim 4.25, there is no such total embedding which extends to the retinues, so either some $z_{m,\rho}$ tag is never placed, or for some ρ extending τ, the retinue of $z_{\sigma^\frown\rho}$ will eventually cease appearing isomorphic to the retinue of the tagged element. Either way, α will have true outcome k for some finite k.

When α takes on outcome k, let m be the size of the largest loop in the temporary label attached to $z_{\sigma^\frown\tau}$ for any (and by Claim 4.32, every) $\sigma \in R_\beta$ for some $\beta \subset \alpha$. Since origin(τ) is incomparable to $\alpha^\frown\langle k\rangle$, it will never be visited again. So this will remain the largest loop size at all future stages. By Claim 4.29, this loop size is not found in the temporary label attached to $z_{\sigma^\frown\tau'}$, but by construction it is in the temporary label attached to b_0, contradicting $\mathcal{B}_\ell \cong \mathcal{A}$. □

Claim 4.34. *If a Φ_ℓ-strategy α is along the true path, $\mathcal{B}_\ell \cong \mathcal{A}$, and a provisional tag is in the correct place at stage s or is placed on a sibling of the correct place, then it is never moved.*

Furthermore, if α is considering a correct b as part of case (3) *of placing a provisional tag, then it never discards that b.*

Proof. Suppose otherwise. It suffices to consider tags of the form $z_{m,\langle j\rangle}$ for some j, as tags of the form $z_{m,\tau}$ with $|\tau| > 1$ only move if the tag on their parent moves, and if $z_{m,\tau}$ is correctly placed, then the tag on its parent is correctly placed.

If a provisional tag $z_{m,\langle j\rangle}$ is correctly placed on an element b or is placed on a sibling of b, then no non-provisional tag can occur in the connected component of b. So the only reason the tag might move is that some predecessor of b gains a provisional tag. By a well-foundedness argument, let b_0 be the earliest element in the tree of b which gains a provisional tag, and let $z_{n,\langle j\rangle}$ be the provisional tag. Then this tag is incorrectly placed, but never moves. This contradicts Claim 4.33.

If α is considering a correct b as part of case (3) of placing a provisional tag, the only reason it would discard that b is if b or some predecessor gains a provisional tag. The argument then proceeds as above. □

Claim 4.35. *If a Φ_ℓ-strategy α is along the true path and $\mathcal{B}_\ell \cong \mathcal{A}$, then a provisional tag can never be placed on a sibling of the correct place.*

Proof. By Claim 4.33 and Claim 4.34. □

Claim 4.36. *If a Φ_ℓ-strategy α is along the true path and $\mathcal{B}_\ell \cong \mathcal{A}$, then for every provisional tag $z_{n,\tau}$ that α seeks to place, there is a stage at which $z_{n,\tau}$ is correctly placed.*

Proof. By induction on the length of τ, where the tag in question is $z_{n,\tau}$.

Let $\{\rho_0, \dots, \rho_{d-1}\}$ be the collection of strings of length n in $\bigcup_{\beta\subset\alpha} R_\beta$. By Claim 4.33 (for the base case $|\tau| = 1$) or the inductive hypothesis, α will eventually be searching the children of $\{f(z_{\rho_0{}^\frown\tau^-}), \dots, f(z_{\rho_{d-1}{}^\frown\tau^-})\}$ to place the tags.

By Claim 4.34, these elements will not be discarded once they come under consideration.

Once these elements are selected for consideration, let m be the loop size currently being searched for. Until the tag $z_{n,\tau}$ is placed, α will never have outcome ∞. Thus the temporary labels will never grow, and so this size will remain unchanged until the tag is placed. This size does occur on $z_{\rho_i \frown \langle j \rangle}$ for every $i < d$; so, since f is an isomorphism, it does occur on a child of $f(z_{\rho_i})$ for every $i < d$, and so the tag will eventually be placed. By Claim 4.35, this tag is correctly placed. □

Claim 4.37. *If a Φ_ℓ-strategy α is along the true path, and $\mathcal{B}_\ell \cong \mathcal{A}$, then α has true outcome ∞.*

Proof. Suppose not. Let k be the true outcome of α. Then by Claim 4.31 and Claim 4.36, there is some stage s when all the tags are in the correct place and α has outcome k. If α is charged with assigning a tag for σ, then the temporary label attached to z_σ cannot grow until α has outcome ∞ (and maybe not even then), which by assumption will never happen again. Thus all these labels are finite. Since $\mathcal{B}_\ell \cong \mathcal{A}$, eventually the corresponding labels in $\mathcal{B}_\ell$ will appear isomorphic, at which point α will have outcome ∞, contradicting our choice of k. □

Claim 4.38. *If a Φ_ℓ-strategy α is along the true path, and $\mathcal{B}_\ell \cong \mathcal{A}$, then the unique isomorphism $f : \mathcal{A} \to \mathcal{B}_\ell$ is computable.*

Proof. Fixing n, let $R^n := \bigcup_{\beta \subset \alpha} R_\beta$, where β ranges over all such $\Xi_{n,j}$-strategies. Note that R^n is finite, and is empty for all but finitely many n.

Non-uniformly, fix $f(z_\sigma)$ for every $\sigma \in R^n$. For every other $\sigma \in T_n$, wait until α has dispensed a tag for z_σ and the tagged element is an immediate successor of $f(z_{\sigma^-})$ (if $|\sigma| > 1$). By Claim 4.31 or 4.36, eventually this will happen. By Claim 4.31 or 4.35, the element so tagged is $f(z_\sigma)$. Thus f restricted to these elements z_σ is computable.

For a point x in the retinue of some σ, wait until $f(z_\sigma)$ is defined. Then x is a part of a loop of some size m attached to z_σ, and this loop is unique in having this size. Wait until a loop of size m appears attached to $f(z_\sigma)$ and define the obvious map.

So f restricted to the retinue of T_n is computable for each n, and furthermore, this is uniformly so for all but finitely many n. Thus f is computable on $\mathcal{A}$. □

This completes the proof of Theorem 1.14. □

5. Index Sets, Computable Categoricity, and Relative Computable Categoricity

In this section, we study the complexity of index sets associated with computably categorical structures and relatively computably categorical structures. In particular, we show the index set complexity of relatively computably categorical structures is Σ^0_3-complete. We also show there is a fixed relatively computably categorical structure whose index set is Σ^0_3-complete and a computably categorical structure whose index set is Π^0_1-complete (within $\mathbb{M}$).

Theorem 1.16. *The index set of the relatively computably categorical structures is Σ^0_3-complete.*

Proof. From the equivalence of (1) and (3) in Theorem 1.3, relative computable categoricity is easily seen to be Σ^0_3.

For Σ^0_3-hardness, we make use of (the proof of) Theorem 4.1 of [7]. There, Downey and Montalbán showed that, given a Σ^0_3-set S, there is a uniformly computable sequence $\{\mathcal{V}_i\}_{i\in\omega}$ of vector spaces over $\mathbb{Q}$ such that $\mathcal{V}_i$ is finite-dimensional if and only if $i \in S$. As it is easy to see that the finite-dimensional vector spaces over $\mathbb{Q}$ are relatively computably categorical (any isomorphism is determined by the image of the (finitely many) basis elements) and that the infinite-dimensional vector spaces over $\mathbb{Q}$ are not (relatively) computably categorical, Σ^0_3-hardness follows. □

If $\mathcal{M}$ is any structure, a priori its index set $\{i : \mathcal{M} \cong \mathcal{M}_i\}$ is Σ^1_1 as it may be rather difficult to tell whether or not $\mathcal{M}$ and $\mathcal{M}_i$ are isomorphic. When $\mathcal{M}$ is computably categorical, it is much simpler as it suffices to check the computable isomorphisms.

Proposition 5.1. *If a computable structure $\mathcal{M}$ is computably categorical, then its index set $\{i : \mathcal{M} \cong \mathcal{M}_i\}$ is Σ^0_3.*

Proof. It suffices to note that $\mathcal{M} \cong \mathcal{M}_i$ if and only if there is an index e such that φ_e is an isomorphism between $\mathcal{M}$ and $\mathcal{M}_i$, i.e., such that φ_e is total, injective, surjective, and preserves the atomic diagram. These are Π^0_2, Π^0_1, Π^0_2, and Π^0_1, respectively. □

Surprisingly, it is rather difficult to find a particular computably categorical structure $\mathcal{M}$ whose index set is Σ^0_3-hard. Natural candidates such as dense linear orders, equivalence structures with classes all of some fixed size, and infinite-dimensional vector spaces over a fixed finite field all have Π^0_2-index sets. Generalizing these examples slightly, any fixed computably categorical linear order, equivalence structure, or vector space has an index set of the complete 1-degree of d.c.e. sets over $\mathbf{0}'$.

Torsion-free abelian groups of rank 1 (or, equivalently, subgroups of the rationals) do provide an example of a (relatively) computably categorical structure $\mathcal{M}$ whose index set is Σ^0_3-hard. The only algebraic background we require is Baer's Theorem, which can be found in any standard reference (see, e.g., Fuchs [8, 9]).

Theorem 5.2 (with Alexander Melnikov)**.** *Let $\mathcal{G}$ be the subgroup of $(\mathbb{Q} : +)$ generated by the set $\{\frac{1}{p} : p$ a prime$\}$. Then $\mathcal{G}$ is relatively computably categorical, and its index set $\{i : \mathcal{G} \cong \mathcal{G}_i\}$ is Σ^0_3-complete.*

Proof. We note $\mathcal{G}$ is relatively computably categorical. For if $\mathcal{H}_1$ and $\mathcal{H}_2$ are presentations of $\mathcal{G}$, an isomorphism $f : \mathcal{H}_1 \to \mathcal{H}_2$ can be defined by fixing a nonzero element $a \in H_1$ and its image $b \in H_2$ under a classical isomorphism. Then to define $f(x)$ for an arbitrary $x \in H_1$, it suffices to search for the rational number q such that $x = qa$, search for the element $y \in H_2$ such that $y = qb$, and let $f(x) := y$. This is readily seen to be an isomorphism and is computable from $\deg(\mathcal{H}_1) \vee \deg(\mathcal{H}_2)$.

We also note that the index set $\{i : \mathcal{G} \cong \mathcal{G}_i\}$ is Σ^0_3 by Proposition 5.1. We show this index set is Σ^0_3-hard by building, for every i, a c.e. subgroup $\mathcal{G}_i \leq (\mathbb{Q}, +)$ such that $\mathcal{G} \cong \mathcal{G}_i$ if and only if $i \in \mathrm{Cof}$. We then exploit the fact that from an index for a c.e. subgroup of a computable group, one can effectively obtain an index for an isomorphic computable group.

Construction: Fix a computable relation $R(i, x, y)$ satisfying

$$i \in \mathrm{Cof} \quad \text{if and only if} \quad \exists x \exists^\infty y \ R(i, x, y).$$

Let $P \subset \omega$ be the set of primes. We build a co-c.e. set $A_i \subseteq P$ in stages, letting $A_{i,s} = \{a_0^s < a_1^s < \dots\}$ be its stage s approximation. At stage s, if $R(i, x, s)$ holds for some $x < s$, we choose x least such, and remove a_x^s from A_i. We enumerate $1/a_x^s$ into $\mathcal{G}_i$. We also close $\mathcal{G}_i$ under the group operations.

Verification: We argue that $\mathcal{G} \cong \mathcal{G}_i$ if and only if $i \in \mathrm{Cof}$. If $i \in \mathrm{Cof}$, choose x least such that $\exists^\infty y \ R(i, x, y)$. Then clearly $|A_i| = x$. Thus $\mathcal{G}_e$ is the subgroup generated by $\{\frac{1}{p} : p \notin \{a_0, \dots, a_{x-1}\}\}$. By Baer's Theorem, this is isomorphic to $\mathcal{G}$.

If instead $i \notin \mathrm{Cof}$, then for every x, let y_x be least such that for all $y > y_x$ and $x' \leq x$, the predicate $R(i, x', y)$ does not hold. Then $a_x = a_x^{y_x}$. Thus $|A_i| = \infty$, and so $\mathcal{G} \ncong \mathcal{G}_i$ by Baer's Theorem. □

Analyzing the opposite extreme, it is natural to ask how simple the index set of a computably categorical structure can be. As determining whether an index is a presentation of a structure can introduce artificial complexity, we restrict ourselves to the class of nonempty computable models.

Definition 5.3. For any signature $\mathcal{L}$, denote the class of all nonempty computable models with signature $\mathcal{L}$ by $\mathbb{M} = \mathbb{M}_{\mathcal{L}}$. Note that we are taking a computable structure to be a computable subset of ω with computable functions, relations, and constants (total on the universe).

Proposition 5.4 (with Adam Day)**.** *There is an infinite computably categorical structure $\mathcal{M}$ whose index set $\{i : \mathcal{M} \cong \mathcal{M}_i\}$ is Π^0_1-complete within $\mathbb{M}$.*

Proof. The signature $\mathcal{L}$ for our structure $\mathcal{M}$ has a unary function S, a binary function f, and constants 0 and 1. The unary function S will be the successor function. The binary function f will satisfy $(\forall i \in M)(\forall s \in M) \left[f(i, s) \in \{0^{\mathcal{M}}, 1^{\mathcal{M}}\}\right]$. The structure $\mathcal{M} := (M : S, f, 0, 1)$ will be such that the reduct $(M : S, 0, 1)$ is (isomorphic to) the standard model $(\omega : S, 0, 1)$, where S is the successor function. The function f will be used to ensure that an expansion $\mathcal{N} = (N : S, f, 0, 1)$ of the theory of $(\omega : S, 0, 1)$ with a nonstandard universe can be easily distinguished as being nonisomorphic to $\mathcal{M}$. In particular, the construction will exploit our working within $\mathbb{M}$ by building the structure $\mathcal{M}$ so that if $\mathcal{M}_i$ is to be isomorphic to $\mathcal{M}$, it must witness any element of itself being standard in an effectively bounded length of time.

Fix an enumeration $\{\mathcal{M}_i\}_{i \in \omega}$ of all presentations of (candidate) structures in the signature $\mathcal{L}$. We denote by $\bar{n}^{\mathcal{M}}$ and $\bar{n}^{\mathcal{M}_i}$ the elements $(S^n(0))^{\mathcal{M}}$ and $(S^n(0))^{\mathcal{M}_i}$, respectively. We assume that at stage s, the element $\overline{(s+1)}^{\mathcal{M}_i}$ is not yet defined. Note that it will be the case that $n = \bar{n}^{\mathcal{M}}$, and that $\bar{n}^{\mathcal{M}_i}$ may not exist.

Construction: The universe M of $\mathcal{M}$ is ω. As already suggested, we define $0^{\mathcal{M}}$ and $1^{\mathcal{M}}$ to be $0 \in \omega$ and $1 \in \omega$, respectively, and define $S(n) = n + 1$ for each $n \in \omega$.

At each stage s, we define $f(i, s)$ for all $i \in \omega$. At stage $s = 0$, we define $f(i, 0) = 0$ for all $i \in \omega$. At stage $s > 0$, the definition of $f(i, s)$ is, by default, the value $f(i, s - 1)$. The exception occurs if the structure $\mathcal{M}_i$ is *challenging* the structure $\mathcal{M}$ at stage s, namely, when $\bar{\imath}^{\mathcal{M}_i}$ and an element x exist in M_i such that $f^{\mathcal{M}_i}(\bar{\imath}^{\mathcal{M}_i}, x)$ is defined but x has not yet been seen to be standard. In this case, let x be the Gödel least such; the definition of $f(i,s)$ is 1 if $f^{\mathcal{M}_i}(\bar{\imath}^{\mathcal{M}_i}, x) = 0^{\mathcal{M}_i}$ and 0 otherwise, and we say that $f^{\mathcal{M}}$ is *accepting the challenge by* (i,x) *starting at stage* s. If x is later seen to be standard in $\mathcal{M}_i$ at a stage t, then we say $\mathcal{M}_i$ *defeated the challenge by* (i,x) *at stage* t. Until the challenge by (i,x) is defeated, $f^{\mathcal{M}}$ does not accept a challenge from any (i,y) with $x \neq y$. If the challenge is defeated, then $f^{\mathcal{M}}$ accepts the challenge from the next Gödel least element from $\mathcal{M}_i$ that presents a challenge.

Verification: By construction, the structure $\mathcal{M}$ is infinite and computable. Moreover, it is computably categorical as a consequence of ω under successor being computably categorical. It therefore suffices to argue that the set

$$\{i : \mathcal{M} \cong \mathcal{M}_i\}$$

is Π^0_1 in $\mathbb{M}$. Fix an index i. As we are working in $\mathbb{M}$, we may assume $S^{\mathcal{M}_i}$ and $f^{\mathcal{M}_i}$ are total on M_i. For each stage s, we believe $\mathcal{M}$ and $\mathcal{M}_i$ are isomorphic if and only if

(1) there is no "trivial reason" to believe otherwise, i.e., $S^{\mathcal{M}_i}$ must appear to be a successor function on M_i, $f^{\mathcal{M}_i}$ must take the value $0^{\mathcal{M}_i}$ or $1^{\mathcal{M}_i}$, $0^{\mathcal{M}_i}$ must not be the successor of any element in M_i, $1^{\mathcal{M}_i}$ must be $S(0^{\mathcal{M}_i})$, and $f^{\mathcal{M}_i}(\bar{\imath}^{\mathcal{M}_i}, \bar{n}^{\mathcal{M}_i})$ must equal $f^{\mathcal{M}}(i,n)$, and
(2) if $f^{\mathcal{M}}$ accepts the challenge by (i,x) starting at stage s, then we have $x \in \left\{\bar{0}^{\mathcal{M}_i}, \bar{1}^{\mathcal{M}_i}, \ldots, \bar{s}^{\mathcal{M}_i}\right\}$.

We argue that if $\mathcal{M} \not\cong \mathcal{M}_i$ then there is some stage after which we believe $\mathcal{M}$ and $\mathcal{M}_i$ are not isomorphic. If $\mathcal{M}$ and $\mathcal{M}_i$ are not isomorphic for a trivial reason, then we will eventually cease believing them to be isomorphic. If $\mathcal{M}$ and $\mathcal{M}_i$ are not isomorphic for a nontrivial reason, then the structure $\mathcal{M}_i$ must have nonstandard elements. So there is some nonstandard element x for which $f^{\mathcal{M}}$ accepted the challenge by (i,x) at some stage s, but $\mathcal{M}_i$ did not defeat the challenge by (i,x). Then, once the elements $\left\{\bar{0}^{\mathcal{M}_i}, \bar{1}^{\mathcal{M}_i}, \ldots, \bar{s}^{\mathcal{M}_i}\right\}$ are defined, we cease believing $\mathcal{M}$ and $\mathcal{M}_i$ to be isomorphic by our definition of $f(i,s)$.

Conversely, if we believe that $\mathcal{M}$ and $\mathcal{M}_i$ are not isomorphic at some stage, there are two possibilities. If we believe them not isomorphic for a trivial reason, then certainly $\mathcal{M} \not\cong \mathcal{M}_i$. If we believe them not isomorphic because a challenge by some (i,x) was accepted starting at stage s and $x \notin \left\{\bar{0}^{\mathcal{M}_i}, \bar{1}^{\mathcal{M}_i}, \ldots, \bar{s}^{\mathcal{M}_i}\right\}$, then there are two cases. If x is a non-standard element of $\mathcal{M}_i$, then $\mathcal{M} \not\cong \mathcal{M}_i$. If x is a standard element of $\mathcal{M}_i$, then let t be the stage at which the challenge was defeated. Then $x \in \left\{\overline{(s+1)}^{\mathcal{M}_i}, \ldots, \bar{t}^{\mathcal{M}_i}\right\}$. But by construction, $f^{\mathcal{M}}(i,n) \neq f^{\mathcal{M}_i}(i,x)$ for all $n \in \left\{\overline{(s+1)}^{\mathcal{M}_i}, \ldots, \bar{t}^{\mathcal{M}_i}\right\}$. Thus $\mathcal{M} \not\cong \mathcal{M}_i$.

We conclude that the index set of $\mathcal{M}$ is Π^0_1 in $\mathbb{M}$. We observe that it is easily seen to be Π^0_1-complete: For a Π^0_1 formula $(\forall s)\,[\varphi(n,s)]$, construct a structure $\mathcal{M}_n$ by copying $\mathcal{M}$ until an s with $\neg\varphi(n,s)$ is seen. At this time, make a "wrong" definition of $f^{\mathcal{M}_n}$, but preserve totality. $\square$

6. Relative Categoricity Above a Degree $\mathbf{d}$

The notions of computable categoricity and relative computable categoricity are traditionally relativized (as in Definition 1.6) by allowing oracle access to a fixed number of jumps over the presentations of the relevant models. Another method of relativization would be to allow oracle access to a fixed degree. We explore this idea in this section. As the constructions are not particularly difficult and introduce no significant new ideas (relying only on the pushing on isomorphisms machinery), we only sketch their proofs.

Definition 6.1. Let $\mathbf{d}$ be a Turing degree. A computable structure $\mathcal{S}$ is *relatively computably categorical above* $\mathbf{d}$ (or *relatively* Δ^0_α*-categorical above* $\mathbf{d}$, respectively) if between any two presentations $\mathcal{A}$, $\mathcal{B} \geq_T \mathbf{d}$ of $\mathcal{S}$, there is an isomorphism computable in $\deg(\mathcal{A}) \cup \deg(\mathcal{B})$ (or $\Delta^0_\alpha(\deg(\mathcal{A}) \cup \deg(\mathcal{B}))$, respectively).

Proposition 6.2. *For a computable structure* $\mathcal{S}$*, the following are equivalent:*

1. *The structure* $\mathcal{S}$ *is relatively* Δ^0_α*-categorical above* $\mathbf{d}$.
2. *Between any two presentations* $\mathcal{A}$ *and* $\mathcal{B}$ *of* $\mathcal{S}$*, there is an isomorphism computable in* $\Delta^0_\alpha(\deg(\mathcal{A}) \cup \deg(\mathcal{B}) \cup \mathbf{d})$.

Proof. If $\mathcal{S}$ is trivial, i.e., if there is a tuple of elements such that every permutation of the universe that fixes this tuple pointwise is an automorphism, then the equivalence is immediate. We therefore assume $\mathcal{S}$ is nontrivial.

For (1) implies (2), we use that the degree spectrum of a structure is upwards closed (see Theorem 3.2 of [2]). From this, we have a presentation $\mathcal{A}'$ and isomorphism $g_1 : \mathcal{A} \to \mathcal{A}'$ with $\deg(\mathcal{A}') = \deg(\mathcal{A}) \cup \mathbf{d}$ and $\deg(g_1) \leq \deg(\mathcal{A}) \cup \mathbf{d}$; and a presentation $\mathcal{B}'$ and isomorphism $g_2 : \mathcal{B} \to \mathcal{B}'$ with $\deg(\mathcal{B}') = \deg(\mathcal{B}) \cup \mathbf{d}$ and $\deg(g_2) \leq \deg(\mathcal{B}) \cup \mathbf{d}$. By relative Δ^0_α-categoricity above $\mathbf{d}$, there is an isomorphism $f : \mathcal{A}' \cong \mathcal{B}'$ with $f \in \Delta^0_\alpha((\deg(\mathcal{A}) \cup \mathbf{d}) \cup (\deg(\mathcal{B}) \cup \mathbf{d}))$. Then $g_2^{-1} \circ f \circ g_1 : \mathcal{A} \cong \mathcal{B}$ is an isomorphism and $\deg(g_2^{-1} \circ f \circ g_1) \leq \Delta^0_\alpha(\deg(\mathcal{A}) \cup \deg(\mathcal{B}) \cup \mathbf{d})$.

The direction (2) implies (1) is immediate. $\square$

For some classes of structures, there is no difference between relative computable categoricity and relative computable categoricity above $\mathbf{d}$ (for any degree $\mathbf{d}$).

Theorem 6.3. *A linear order is relatively computably categorical above a degree* $\mathbf{d}$ *if and only if it is relatively computably categorical.*

A Boolean algebra is relatively computably categorical above a degree $\mathbf{d}$ *if and only if it is relatively computably categorical.*

Proof. The proof that a (relatively) computably categorical linear order can possess at most finitely many adjacencies succeeds in the presence of a $\mathbf{d}$-oracle, as does the proof that a (relatively) computably categorical Boolean algebra can possess at most finitely many atoms. $\square$

On the other hand, there are classes of structures where this notion does not coincide with either computable categoricity or relative computable categoricity.

Theorem 6.4. *For any nonzero c.e. degree* $\mathbf{d}$*, there is a structure* $\mathcal{S}$ *that is relatively computably categorical above* $\mathbf{d}$ *but not computably categorical.*

Proof. Fix a c.e. set $D \in \mathbf{d}$. The structure $\mathcal{S}$ we construct is a rigid undirected graph.

Construction: The isomorphism type of $\mathcal{S}$ contains an "ω-spine", i.e., a sequence of vertices in order type ω. Emanating from the nth element of the spine is a path of length 1 and a path of length 2 if $n \notin D$, and a path of length 2 and a path of length 3 if $n \in D$.

Verification: Towards showing that $\mathcal{S}$ is relatively computably categorical above $\mathbf{d}$, fix presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{S}$. We show how $\deg(\mathcal{A}) \cup \deg(\mathcal{B}) \cup \mathbf{d}$ computes an isomorphism $f : \mathcal{A} \to \mathcal{B}$. We non-uniformly know the initial elements of the spines in $\mathcal{A}$ and $\mathcal{B}$. The function f maps the ω-spines in the obvious way, noting the ω-spines of $\mathcal{A}$ and $\mathcal{B}$ can be effectively found using $\deg(\mathcal{A})$ and $\deg(\mathcal{B})$. For the nth element of the spine, if $n \in D$, the function f waits until both a path of length 2 and a path of length 3 appear in both $\mathcal{A}$ and $\mathcal{B}$. Then it maps them as appropriate. If $n \notin D$, the function f waits until both a path of length 1 and a path of length 2 appear in both $\mathcal{A}$ and $\mathcal{B}$, mapping them appropriately. It is clear that f is an isomorphism computable in $\deg(\mathcal{A}) \cup \deg(\mathcal{B}) \cup \mathbf{d}$.

Towards showing that $\mathcal{S}$ is not computably categorical, we exhibit computable copies $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{S}$ that are not computably isomorphic. For $\mathcal{A}$, we construct the ω-spine with a path of length 1 and a path of length 2 at every $n \in \omega$. When we see a number n enter D, we extend the path of length 1 at the nth element of the ω-spine to be a path of length 3. For $\mathcal{B}$, we construct the ω-spine with a path of length 1 and a path of length 2 at every $n \in \omega$. When we see a number n enter D, we extend the path of length 1 at the nth element of the ω-spine to be a path of length 2 and extend the path of length 2 at the nth element of the ω-spine to be a path of length 3. The unique isomorphism $\pi : \mathcal{A} \to \mathcal{B}$ computes $\mathbf{d}$ as membership of n in D can be determined by noting whether the initial path of length 1 in $\mathcal{A}$ is mapped to the initial path of length 1 in $\mathcal{B}$ (in which case $n \notin D$) or not (in which case $n \in D$). □

Remark 6.5. We note that Theorem 6.4 can easily be improved to all nonzero d.c.e. degrees by exploiting the structures introduced by Csima, Franklin, and Shore in [5].

Theorem 6.6. *For any nonzero c.e. degree* $\mathbf{d}$*, there is a computable structure* $\mathcal{S}$ *that is computably categorical, relatively computably categorical above* $\mathbf{d}$*, but not relatively computably categorical.*

Proof. Fix a c.e. set $D \in \mathbf{d}$. The structure $\mathcal{S}$ is again an undirected graph containing an ω-spine with two finite paths emanating from each vertex of the ω-spine. As in Theorem 6.4, we attempt to increase the lengths of the paths emanating from an element of the ω-spine when n enters D. Here, however, we must respect the pushing on isomorphism machinery: If n enters D, we immediately increase the path of length two to a path of length three; we do not increase the path of length one to a path of length two until the higher priority isomorphism requirements permit. Unlike in the proof of Theorem 3.1, diagonalization strategies do not claim a location to work at until they are ready to act.

Construction: We construct a computable presentation $\mathcal{A}$, taking $\mathcal{S}$ to be its isomorphism type. The structure $\mathcal{A}$ contains an ω-spine. Emanating from the nth element of the spine is a path of length one and a path of length two if $n \notin D$. If and when n enters D, we extend the path of length two at the nth element of the spine to a path of length three. Let a_n be the first element in the path of original length two (i.e., the element which is adjacent to the nth element of the spine).

As in the proof of Theorem 3.1, we have a tree of strategies, some constructing isomorphisms and some diagonalizing against Scott families. A strategy σ attempting to defeat a Scott family $(\overline{c}_i, X_i)$ of existential formulas is *ready to act* at stage s if:

(1) The strategy σ has not already acted.
(2) The strategy σ is accessible at stage s.

(3) There is some $n \in D_s$, a previous stage $t < s$, and some $\varphi \in X_i$ such that:
- $n \notin D_t$,
- $\mathcal{A}_t \models \varphi(a_n, \overline{c}_i)$, and
- the parameter $\overline{c}_i$ is disjoint from the paths emanating from the nth element of the spine.

A strategy that is ready to act acts by growing the path of length one emanating from the nth element of the spine into a path of length two (if some other strategy has not already done this).

Verification: As we build a computable structure, it is immediate that we have a computably categorical structure as a consequence of the usage of the pushing on isomorphism machinery. We therefore verify that it is not relatively computably categorical and that it is relatively computably categorical above $\mathbf{d}$.

Clearly if some strategy σ working to defeat $(\overline{c}_i, X_i)$ acts at some stage, then $(\overline{c}_i, X_i)$ cannot be a Scott family for $\mathcal{S}$: The formula φ holds of an element in the path of length 3 and of an element in the path of length 2, despite the structure being rigid. So if $(\overline{c}_i, X_i)$ is a Scott family, then the strategy along the true path working to defeat it never acts. Consequently, for any $n \in D$ and $\varphi \in X_i$, we have that $\mathcal{A}_s \models \varphi(a_n, \overline{c}_i)$ only if $n \in D_s$ or some element of $\overline{c}_i$ occurs in a path coming out of n. Thus the computable function

$$n \mapsto (\mu s)\,[(\forall m \leq n)(\exists \varphi \in X_{i,s})\,[\mathcal{A}_s \models \varphi(a_m, \overline{c}_i)]]$$

is total and a finite modification of it majorizes the modulus of D, contradicting D being non-computable.

The structure is relatively computably categorical above $\mathbf{d}$ as we can construct a $\deg(\mathcal{A}) \cup \deg(\mathcal{B}) \cup \mathbf{d}$-isomorphism between any two presentations $\mathcal{A}$ and $\mathcal{B}$. Just as in Theorem 6.4, we may non-uniformly map the ω-spines. For the nth element of the ω-spine, we check whether n is in D. If it is, we wait to see a path of length three in both $\mathcal{A}$ and $\mathcal{B}$ before mapping either path; if it is not, we wait only to see a path of length two before mapping either path. In either case, we know that the other path must be shorter (though we do not necessarily know its length), so our mapping cannot be wrong. $\square$

By changing the widgets attached to the elements of the ω-spine, we obtain a similar theorem at the level of one jump higher.

Theorem 6.7. *There is a computable structure $\mathcal{S}$ that is computably categorical, relatively computably categorical above $\mathbf{0}''$, but not relatively Δ^0_2-categorical.*

Proof. The structure $\mathcal{S}$ is again an undirected graph with an ω-spine. Unlike in the proof of Theorem 6.4 and Theorem 6.6, the widgets emanating from the nth element of the spine will be cliques (vertex sets with edges between every two vertices) rather than paths. Depending on the behavior of the strategy controlling n, these cliques will either both be infinite, or of finite sizes k and $k+2$ for some k.

We construct a computable presentation $\mathcal{A}$, taking $\mathcal{S}$ to be its isomorphism type. For any Σ^c_2-formula ψ, the statement "$\mathcal{A} \models \psi(\overline{x}, \overline{c})$" is effectively equivalent to $(\forall^\infty y)\,[\varphi(\overline{x}, \overline{c}, y)]$, for some computable relation φ. We therefore diagonalize against c.e. families of formulas of this form.

Strategy for Defeating a Family $(X_i, \overline{c}_i)$: The strategy is assigned to work with the nth element of the spine, for some n. We begin by constructing a clique of size one and a clique of size three emanating from this element. Let a_n be some point in the larger clique and b_n be some point in the smaller clique.

If σ is accessible at stage s, let $t < s$ be the last stage at which σ was accessible (with $t := 0$ if there is no such stage). Let (r_s, φ_s) be the least pair (by Gödel number) such that $r_s \in \omega$, $(\forall^\infty y)\,[\varphi_s(x, \overline{c}_i, y)] \in X_{i,s}$, and $\varphi_s(a_n, \overline{c}_i, y)$ and $\varphi_s(b_n, \overline{c}_i, y)$ both hold for all y with $r_s \leq y < s$. If $(r_t, \varphi_t) = (r_s, \varphi_s)$, then σ does nothing at stage s. Otherwise, the strategy σ grows each clique by one element (being careful to never use elements of $\overline{c}_i$).

Construction: We place the strategies on a priority tree in the usual fashion, including computable categoricity strategies which use the usual pushing on isomorphism machinery. At every stage, we let all accessible strategies act in order of priority.

Verification: As we build a computable structure, it is immediate that we have a computably categorical structure as a consequence of the usage of pushing on isomorphisms. We therefore verify that it is not relatively Δ^0_2-categorical and that it is relatively computably categorical above $\mathbf{0}''$.

Suppose towards a contradiction that $(\overline{c}_i, X_i)$ is a Scott family of Σ^c_2-formulas. Let n be the number assigned to the strategy along the true path which diagonalizes against $(\overline{c}_i, X_i)$. If there is some formula $\psi(\overline{x}) = (\forall^\infty y)\psi(\overline{x}, y)$ in X_i with $\mathcal{A} \models \psi(a_n, \overline{c}_i) \wedge \psi(b_n, \overline{c}_i)$, then there is some Gödel least pair (r, φ) such that $(\forall y \geq r)[\varphi(a_n, \overline{c}_i, y) \wedge \varphi(b_n, \overline{c}_i, y)]$. Then this pair will be (r_s, φ_s) for all but finitely many s, and thus the two cliques will be of finite, distinct sizes. Thus a_n and b_n will not be in the same orbit, contradicting $(\overline{c}_i, X_i)$ being a Scott family.

If there is no such formula ψ, then for every pair (r, φ), there is some $y > r$ such that at least one of $\varphi(a_n, \overline{c}_i, y)$ or $\varphi(b_n, \overline{c}_i, y)$ fails. So (r, φ) will not be (r_s, φ_s) for any $s > y$. So there are infinitely many stages at which the cliques attached to n grow. So they will be infinite, and thus a_n and b_n will be in the same orbit, contradicting $(\overline{c}_i, X_i)$ being a Scott family.

The structure is relatively computably categorical above $\mathbf{0}''$ because $\mathbf{0}''$ can determine the eventual behavior of the strategy controlling n. Given two copies $\mathcal{A}$ and $\mathcal{B}$, if the two cliques at n are infinite, it does not matter which clique in $\mathcal{A}$ maps to which in $\mathcal{B}$, so a simple back-and-forth argument can construct an isomorphism. If the two cliques are finite, then $\mathbf{0}''$ can determine when they have stopped growing, and then we can wait for appropriately sized cliques in $\mathcal{A}$ and $\mathcal{B}$ before defining our map. □

Theorem 6.8. *There is a structure $\mathcal{S}$ that is computably categorical, relatively Δ^0_2-categorical, and not relatively computably categorical above* $\mathbf{d}$ *for any degree* $\mathbf{d}$.

Proof. The structure built in the proof of Theorem 3.3 suffices. It is computably categorical by construction. Since it is 1-decidable, by Theorem 1.13, it is relatively Δ^0_2-categorical (it is also easy to exhibit a Scott family). For any degree $\mathbf{d} \geq \mathbf{0}'$, the construction of $\mathcal{B}$ can be modified to produce a $\mathbf{d}$-computable structure which is not isomorphic to $\mathcal{A}$ by any $\mathbf{d}$-computable isomorphism. This suffices as, fixing an arbitrary degree $\mathbf{d}$, the structure $\mathcal{S}$ will not be relatively computably categorical above $\mathbf{d} \oplus \mathbf{0}'$, and so not relatively computably categorical above $\mathbf{d}$. □

7. Open Questions

We close by asking various questions that remain open. We start by reiterating questions that were already asked in the introduction.

Question 7.1. What is the index set complexity of the computably categorical structures? Is it arithmetical?

Question 7.2. Is there a computably categorical structure that is not relatively hyperarithmetically categorical?

In recent years, the degree of categoricity of a computable structure has received increasing attention.

Definition 7.3. A computable structure $\mathcal{S}$ has *degree of categoricity* $\mathbf{d}$ if:

(1) Between any two computable presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{S}$, there is a $\mathbf{d}$-computable isomorphism $\pi : \mathcal{A} \cong \mathcal{B}$.
(2) There exist computable presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{S}$ such that every isomorphism $\pi : \mathcal{A} \cong \mathcal{B}$ computes $\mathbf{d}$.

Question 7.4. What is the relationship between the class of structures that are relatively computably categorical above a degree $\mathbf{d}$ and the class of structures having degree of categoricity $\mathbf{d}$?

References

[1] Christopher J. Ash, Categoricity in hyperarithmetical degrees, *Ann. Pure Appl. Logic*, 34(1):1–14, 1987.
[2] Christopher J. Ash and Julia F. Knight, *Computable structures and the hyperarithmetical hierarchy*, volume 144 of *Studies in Logic and the Foundations of Mathematics*, North-Holland Publishing Co., Amsterdam, 2000.
[3] Christopher J. Ash, Julia F. Knight, and Theodore A. Slaman, Relatively recursive expansions. II, *Fund. Math.*, 142(2):147–161, 1993.
[4] Christopher J. Ash and Anil Nerode, Intrinsically recursive relations, *Aspects of Effective Algebra* (John N. Crossley, ed.), Upside Down A Book Co., Yarra Glen, Vic., Australia (1981) 26-41.
[5] Barbara F. Csima, Johanna N.Y. Franklin, and Richard A. Shore, Degrees of Categoricity, in preparation.
[6] Rodney G. Downey, Denis R. Hirschfeldt, and Bakhadyr M. Khoussainov, Uniformity in the theory of computable structures, *Algebra Logika*, 42(5):566–593, 637, 2003.
[7] Rodney G. Downey and Antonio Montalbán, The isomorphism problem for torsion-free abelian groups is analytic complete. *J. Algebra*, 320(6):2291–2300, 2008.
[8] László Fuchs, *Infinite abelian groups. Vol. I*, Pure and Applied Mathematics, Vol. 36, Academic Press, New York, 1970.
[9] László Fuchs, *Infinite abelian groups. Vol. II*, Pure and Applied Mathematics, Vol. 36-II, Academic Press, New York, 1973.
[10] Sergey S. Goncharov, Selfstability, and computable families of constructivizations, *Algebra i Logika*, 14(6):647–680, 727, 1975.
[11] Sergey S. Goncharov, The number of nonautoequivalent constructivizations, *Algebra i Logika*, 16(3):257–282, 377, 1977.
[12] Oleg V. Kudinov, An autostable 1-decidable model without a computable Scott family of $\exists$-formulas, *Algebra i Logika*, 35(4):458–467, 498, 1996.
[13] Robert I. Soare, *Recursively Enumerable Sets and Degrees*, Springer-Verlag, 1987.
[14] Yuri 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.
[15] Walker M. White, On the complexity of categoricity in computable structures, *Math. Log. Q.*, 49(6):603–614, 2003.

Department of Mathematics, Victoria University of Wellington, Wellington, New Zealand
E-mail address: `Rod.Downey@msor.vuw.ac.nz`
URL: `http://msor.victoria.ac.nz/Main/RodDowney`

Department of Mathematics, University of Chicago, Chicago, IL 60637, USA
E-mail address: `asher.kach@gmail.com`

Department of Mathematics, University of Wisconsin, Madison, WI 53706-1388, USA
E-mail address: `lempp@math.wisc.edu`
URL: `http://www.math.wisc.edu/~lempp/`

Department of Mathematics, Victoria University of Wellington, Wellington, New Zealand
E-mail address: `dan@msor.vuw.ac.nz`
URL: `http://msor.victoria.ac.nz/Main/DanTuretsky`