COMPUTABLE CATEGORICITY VERSUS RELATIVE 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. We show that every 1-decidable computably categorical structure is relatively Δ^0_2-categorical. We study the complexity of various index sets associated with computable categoricity and relative computable categoricity. We also 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.

Date: January 4, 2013.

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, Grant # 13407 by the John Templeton Foundation entitled "Exploring the Infinite by Finitary Means", and AMS-Simons Foundation Collaboration Grant 209087. 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 the bulk of this work was carried out. Downey and Lempp would like to thank the Newton Institute for its hospitality during the time the final draft of this paper was prepared. Finally, all the authors would like to thank Andrew Lewis and Antonio Montalbán for helpful corrections on a previous draft, the referee for help comments and corrections, and Adam Day and Alexander Melnikov for allowing us to include Proposition 4.4 and Theorem 4.2, respectively.

[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*.

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 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 [11])**.** *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 (finitary) 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 (finitary) existential formulas.

(3) The structure $\mathcal{S}$ has a c.e. family Φ of (finitary) 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 (finitary) existential formulas.

The program of aligning computational properties of structures with effective syntactic properties goes back to the pioneering work of Goncharov [11], Ash and Nerode [4] and others, and is the theme of Ash and Knight [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 the arithmetic or analytic 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 was resolved by Downey, Kach, Lempp, Lewis, Montalbán, and Turetsky [7] using techniques from this paper.

Theorem 1.5 (Downey, Kach, Lempp, Lewis, Montalbán, Turetsky [7])**.** *The index set of the computably categorical structures is* Π^1_1*-complete.*

Previously, the best-known lower bound was shown by White [17], namely, Π^0_4-hardness. Recently, Hirschfeldt, Kramer, Miller, and Shlapentokh [13] have shown that the index set of the computably categorical algebraic fields is Π^0_4-hard.

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 1.10). 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 level of the hyperarithmetic hierarchy, as will be shown in [7]. This result uses technology introduced here.

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 *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 $\Delta^0_\alpha(\mathcal{B})$.

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

Definition 1.7 (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 1.8 (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 together Theorem 1.3 and Theorem 1.5 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 definition.

Definition 1.9. 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 1.10 (Goncharov [11])**.** *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 [12]; in fact, Kudinov showed that even 1-decidability is not sufficient to ensure computable categoricity and relative computable categoricity coincide.

Theorem 1.11 (Kudinov [14])**.** *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 1.12. *Any 1-decidable, computably categorical structure is relatively Δ^0_2-categorical.*

We will prove Theorem 1.12 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 as evidenced by the following:

Theorem 1.13 (Downey, Kach, Lempp, Lewis, Montalbán, Turetsky [7])**.** *For every computable ordinal α, there is a computably categorical structure that is not relatively Δ^0_α-categorical.*

This suggests the following natural, and important, question.

Question 1.14. 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 4, we prove the following:

Theorem 1.15 (Folklore)**.** *The index set of the relatively computably categorical structures is Σ^0_3-complete.*

In Section 4, we also examine the index set complexity of certain fixed computably categorical and relatively computably categorical structures, such as torsion-free abelian groups and structures with unusual index sets. In Section 5, 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 [15].

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 Lemma 2.2.

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_1^c-type as well as the (classically infinitary) Σ_1^i-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_2^0-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 $F \upharpoonright \overline{b}_0$ 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 $F \upharpoonright \overline{b}_0$ 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 $F_s(\bar{b}) := F_{s-1}(\bar{b}')$ and $F_s(\bar{d}) := \bar{c}$ might be useful. If so, fix some triple and some $i_0 < s$ for which (1) holds. We use 1-decidability to determine whether

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

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 $F_s(\bar{b}) := F_{s-1}(\bar{b}')$ and $F_s(\bar{d}) := \bar{c}$, 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 $F_s(\bar{b}) := F_{s-1}(\bar{b})$ and $F_s(\bar{d}) := F_{s-1}(\bar{d})$, 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 $(\bar{b}, \bar{b}', \bar{d})$ and associated $\bar{c}$ being considered, we use 1-decidability to determine if

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

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

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

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 $F_s(\bar{b}) := F_{s-1}(\bar{b}')$ and $F_s(\bar{d}) := \bar{c}$ 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 $F_s(\bar{b}) := F_{s-1}(\bar{b})$ and $F_s(\bar{d}) := F_{s-1}(\bar{d})$, 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 $\theta_s(\bar{a})$ true about $\mathcal{A}$ and enumerating $\theta_s(F_s(\bar{a}))$ 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 $\bar{b}_0$ be the restraint of σ. We show the desired relationships between the types of tuples of $\mathcal{A}$ using $\overline{p} := F(\bar{b}_0)$.

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 $\overline{b}, \overline{b}' \in \mathrm{dom}(\varphi_{k,s})$. 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 $\overline{b}_0, \overline{b}, \overline{b}', \overline{d} \in \mathrm{dom}(\varphi_{k,s})$. 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 $\exists \overline{x}\chi(\overline{p}\,\overline{a}', \overline{x}) \in \Sigma^c_2\text{-tp}(\overline{p}\,\overline{a}')$ as desired. □

This completes the proof of Lemma 2.2. □

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

Theorem 1.12. *Any 1-decidable, computably categorical structure $\mathcal{A}$ is relatively Δ^0_2-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^c_1-formula.

We show that the family of formulas $\{\chi_{\overline{a}}(\overline{x})\}_{\overline{a} \in A}$ constitutes a Scott family. By Theorem 1.8, it suffices to show that they isolate the Σ^c_2-types. We therefore suppose $\mathcal{A} \models \chi_{\overline{a}}(\overline{a}')$ and show $\Sigma^c_2\text{-tp}_{\overline{p}}(\overline{a}') = \Sigma^c_2\text{-tp}_{\overline{p}}(\overline{a})$. 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 $\Sigma^c_2\text{-tp}_{\overline{p}}(\overline{a}') = \Sigma^c_2\text{-tp}_{\overline{p}}(\overline{a})$.

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^0_2-categorical. □

3. Pushing on Isomorphisms and Results Related to Theorem 1.12

Theorem 1.12 raises questions about various ways in which the hypotheses can be weakened or the conclusion strengthened. In this section, we explore a number of such 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 [12, 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 $\overline{c}_i$ elementwise.

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 $\tau\hat{}\mathtt{wait} \subseteq \sigma$, we consider the root vertices created by τ; for each $\tau \subset \sigma$ such that $\tau\hat{}\mathtt{act} \subseteq \sigma$ 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 $\sigma\hat{}\mathtt{k}$, 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 $\mathtt{k}$.

Construction: For an $\mathcal{S}_j$-strategy, we order the outcomes as $\infty < \cdots < 2 < 1 < 0$. For an $\mathcal{R}_i$-strategy, we order the outcomes as $\mathtt{act} < \mathtt{wait}$. 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 $\tau \supseteq \sigma\hat{}\infty$, if τ has final outcome $\mathtt{wait}$, 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 1.12. One might think that a simpler way to prove it would be to relativize Goncharov's Theorem 1.10. 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 1.10 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 H 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'. □

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 1.12:

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 for the pair 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 build a structure $\mathcal{A}$ and structures $\{\mathcal{B}_i\}_{i\in\omega}$ to meet the following requirements:

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

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

Strategy for meeting $\mathcal{R}_i$: The strategy 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. If X_i is not a true Δ^0_2-set, then $\mathcal{P}_{i,j}$ is trivially satisfied, so we assume that X_i is a true Δ^0_2-set. We argue if $g_j : \mathcal{A} \cong \mathcal{B}_i$ is an isomorphism, then X_i is computable. Let x be the root vertex of the larger component built by the $\mathcal{P}_{i,j}$-strategy along the true path.

We note if $g_j : \mathcal{A} \cong \mathcal{B}_i$ is an isomorphism, then the parameter r for $\mathcal{P}_{i,j}$ grows without bound. For if it was bounded, then $g_j(x)$ and $F_i(x)$ would coincide as x would have a unique image in $\mathcal{B}_i$ as the appropriate components are finite. However this would imply that $g_{j,s}(x)$ and $F_{i,s}(x)$ coincide at infinitely many stages, causing r to grow without bound, contradicting the boundedness of r.

We therefore assume the parameter r for $\mathcal{P}_{i,j}$ grows without bound. For any n, let s_n be the least stage at which the parameter r equals n. Let t be the least stage for which $g_j(x) = g_{j,s}(x)$ for all $s > t$, noting t exists as we are supposing g_j is an isomorphism. For any z, we claim that $X \upharpoonright z = X_{s_n} \upharpoonright z$, where $n > z$ is any number with $s_n > t$ and such that $X_s \upharpoonright z$ was constant for $s \in [s_n, s_{n+1}]$. From this it will follow that X is computable.

To see this, suppose otherwise. Then let $t' > s_n$ be least with $X \upharpoonright z = X_{t'} \upharpoonright z$. Consider those axioms in Γ_i concerning loops attached to $F_i(x)$; note that before stage s_n, no facts concerning any loop of size $n+2$ or greater had been enumerated. So until stage t', every axiom concerning a loop of size $n+2$ or greater had use extending $X_{s_n} \upharpoonright z$. Thus at stage t', all loops attached to $F_i(x)$ of size at least $n+2$ are removed from $\mathcal{B}_i$. By assumption $t' > s_{n+1}$, so at stage t', the largest two loops attached to $F_i(x)$ are both of size at least $n+2$. So at stage t', $\mathcal{R}_i$ has the opportunity to redefine $F_i(x)$, interchanging components. These components might swap back at a later stage due to $X_s \upharpoonright z$ reverting to $X_{s_n} \upharpoonright z$, but at every stage $s > t'$ with $X_s \upharpoonright z = X_{t'} \upharpoonright z$, we will have $F_{i,s}(x) \neq g_{j,t}(x)$, and $g_{j,t}(x) = g_{j,s}(x)$ by choice of t. Since $X_{t'} \upharpoonright z = X \upharpoonright z$, $F_{i,s}(x)$ and $g_{j,s}(x)$ will differ at all but finitely many stages s, contrary to the assumption that r grows without bound. □

This completes the proof of Theorem 3.5. □

4. 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. Though the authors are not aware of any proofs of this fact in the literature, we attribute this result to folklore as it is certainly known to many. 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 1.15 (Folklore)**.** *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 [8]. 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 4.1. *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 [9, 10]).

Theorem 4.2 (with Alexander Melnikov)**.** *Let $\mathcal{G}$ be the subgroup of $(\mathbb{Q} : +)$ generated by the set $\{\frac{1}{p} : p \text{ a prime}\}$. 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 4.1. 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 $A_{i,s} = \{a^s_0 < a^s_1 < \dots\}$ 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^s_x from A_i. We enumerate $1/a^s_x$ 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 $a_x = a^{y_x}_x$. 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 4.3. 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 4.4 (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 $\mathcal{M} \ncong \mathcal{M}_i$ 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 $\mathcal{M} \ncong \mathcal{M}_i$. 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 $\mathcal{M} \ncong \mathcal{M}_i$. 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. □

5. Relative Categoricity Above a Degree

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 5.1. 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 5.2. *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 3.21 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. □

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 5.3. *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. □

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

Theorem 5.4. *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. For each n, two paths emanate from the nth element of the spine: If $n \notin D$, the paths have lengths 1 and 2; while if $n \in D$, the paths have lengths 2 and 3. Clearly, this $\mathcal{S}$ is computably presentable.

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 5.5. We note that Theorem 5.4 can be improved significantly by exploiting the structures introduced by Csima, Franklin, and Shore [5]. Indeed, the result remains true for any degree $\mathbf{d}$ that is d.c.e. and above some $\mathbf{0}^{(\alpha)}$, where α is a nonlimit computable ordinal.

Theorem 5.6. *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 5.4, 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

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

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 5.4, 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 5.7. *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 5.4 and Theorem 5.6, 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 5.8. *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 1.12, 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 $\mathbf{d}$. □

References

[1] Christopher J. Ash, Categoricity in hyperarithmetical degrees, *Annals of Pure and Applied 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, *Fundamenta Mathematicae*, 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 and the Hyperarithmetic Hierarchy, *Notre Dame Journal of Formal Logic*, to appear.

[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, Asher M. Kach, Steffen Lempp, Andrew E. M. Lewis, Antonio Montalbán, and Daniel D. Turetsky, The complexity of computable categoricity, in preparation.

[8] Rodney G. Downey and Antonio Montalbán, The isomorphism problem for torsion-free abelian groups is analytic complete. *Journal of Algebra*, 320(6):2291–2300, 2008.

[9] László Fuchs, *Infinite abelian groups. Vol. I*, Pure and Applied Mathematics, Vol. 36, Academic Press, New York, 1970.

[10] László Fuchs, *Infinite abelian groups. Vol. II*, Pure and Applied Mathematics, Vol. 36-II, Academic Press, New York, 1973.

[11] Sergey S. Goncharov, Autostability, and computable families of constructivizations, *Algebra i Logika*, 14(6):647–680, 727, 1975.

[12] Sergey S. Goncharov, The number of nonautoequivalent constructivizations, *Algebra i Logika*, 16(3):257–282, 377, 1977.

[13] Denis R. Hirschfeldt, Kenneth L. Kramer, Russell G. Miller, and Alexandra Shlapentokh, Categoricity properties for computable algebraic fields, submitted.

[14] 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.

[15] Robert I. Soare, *Recursively Enumerable Sets and Degrees*, Springer-Verlag, 1987.

[16] 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.

[17] Walker M. White, On the complexity of categoricity in computable structures, *Mathematical Logic Quarterly*, 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://homepages.msor.vuw.ac.nz/~downey/

E-mail address: asher.kach@gmail.com
URL: http://www.math.uchicago.edu/~kach/

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/

Kurt Gödel Research Center
E-mail address: turetsd4@univie.ac.at
URL: http://tinyurl.com/dturetsky