Degrees that are low for isomorphism

Johanna N.Y. Franklin
University of Connecticut
Reed Solomon
University of Connecticut

September 12, 2013

Abstract

We say that a degree is low for isomorphism if, whenever it can compute an isomorphism between a pair of computable structures, there is already a computable isomorphism between them. We show that while there is no clear-cut relationship between this property and other properties related to computational weakness, the low-for-isomorphism degrees contain all Cohen 2-generics and are disjoint from the Martin-Löf randoms. We also consider lowness for isomorphism with respect to the class of linear orders.

1 Introduction

Within classical computability theory, there are many ways to specify that a particular set A or Turing degree $\mathbf{d}$ is computationally weak. For example, minimal degrees, low degrees and hyperimmune-free degrees are each computationally weak in an appropriate sense. More recently, there has been considerable interest in sets (or degrees) which are low for $\mathcal{P}$ for various relativizable notions $\mathcal{P}$. Roughly, a set A is low for $\mathcal{P}$ if the relativized notion $\mathcal{P}^A$ is the same as $\mathcal{P}$. For example, A is low for Martin-Löf randomness if the collection of sets which are Martin-Löf random relative to A is the same collection of sets which are Martin-Löf random. (See [7] and Chapter 5 of Nies [13] for additional examples and motivation.)

In this paper, we examine a lowness notion in computable model theory which is related to the study of degrees of categoricity. We begin with a summary of definitions from computable model theory to fix our notation.

For a degree $\mathbf{d}$ and computable structures $\mathcal{A}$ and $\mathcal{B}$, we say $\mathcal{A}$ *is* $\mathbf{d}$*-computably isomorphic to* $\mathcal{B}$, denoted $\mathcal{A} \cong_{\mathbf{d}} \mathcal{B}$, if there is an isomorphism between $\mathcal{A}$ and $\mathcal{B}$ which is computable from $\mathbf{d}$. If $\mathbf{d} = \mathbf{0}$, we say $\mathcal{A}$ *is computably isomorphic to* $\mathcal{B}$ and write $\mathcal{A} \cong_{\Delta^0_1} \mathcal{B}$. A computable structure $\mathcal{A}$ is $\mathbf{d}$-*computably categorical* if for every computable structure $\mathcal{B}$ which is classically isomorphic to $\mathcal{A}$, we have $\mathcal{A} \cong_{\mathbf{d}} \mathcal{B}$.

Definition 1.1. A degree $\mathbf{d}$ is a *degree of categoricity* if there is a computable structure $\mathcal{A}$ such that $\mathcal{A}$ is $\mathbf{c}$-computably categorical if and only if $\mathbf{c} \geq \mathbf{d}$.

Fokina, Kalimullin and Miller [6] introduced degrees of categoricity. They showed that every degree which is d.c.e. in and above $\mathbf{0^{(n)}}$ is a degree of categoricity and that $\mathbf{0}^{(\omega)}$ is a degree of categoricity. Csima, Franklin and Shore [4] extended these results to show that for every computable ordinal α, $\mathbf{0}^{(\alpha)}$ is a degree of categoricity and for every computable successor ordinal α, each degree d.c.e. in and above $\mathbf{0}^{(\alpha)}$ is a degree of categoricity. In the negative direction, Csima, Franklin and Shore proved that every degree of categoricity is hyperarithmetic and hence that there are only countably many degrees of categoricity.

Anderson and Csima [1] continued working in a negative direction and developed several methods to show that certain types of degrees are not degrees of categoricity. In particular, they gave an alternate proof that there are only countably many degrees of categoricity and proved that every noncomputable hyperimmune-free degree is not a degree of categoricity.

More importantly for our current work, Anderson and Csima gave an oracle construction of a noncomputable degree $\mathbf{d} \leq \mathbf{0}''$ which is not a degree of categoricity by showing that for every pair of isomorphic computable structures $\mathcal{A}$ and $\mathcal{B}$, if $\mathcal{A} \cong_{\mathbf{d}} \mathcal{B}$, then $\mathcal{A} \cong_{\Delta^0_1} \mathcal{B}$. Such a degree $\mathbf{d}$ is computationally weak in the sense that $\mathbf{d}$ can only tell that two computable structures are isomorphic when these structures are in fact computably isomorphic. We isolate this property and refer to such degrees as being low for isomorphism.

Definition 1.2. A degree $\mathbf{d}$ is *low for isomorphism* if for every pair of computable structures $\mathcal{A}$ and $\mathcal{B}$, $\mathcal{A} \cong_{\mathbf{d}} \mathcal{B}$ if and only if $\mathcal{A} \cong_{\Delta^0_1} \mathcal{B}$.

Anderson and Csima's oracle construction of a low-for-isomorphism degree can be recast as a forcing construction. In Section 2, we use three different forcing notions to construct low-for-isomorphism degrees and compare these degrees with other notions of computationally weak degrees. In the cases of Mathias forcing and Cohen forcing, the fact that sufficiently generic degrees are low for isomorphism follows from work by Hirschfeldt and Shore [9] and Hirschfeldt, Shore and Slaman [10]. The fact that Sacks forcing also produces low-for-isomorphism degrees does not appear to be in the literature, but it is a minor modification and has been observed by several people. We include a proof here for the sake of completeness.

In Section 3, we give examples of degrees which are not low for isomorphism. In particular, we show that if $\mathbf{d}$ can compute a noncomputable Δ^0_2 degree or can compute a separating set for a pair of computably inseparable c.e. sets, then $\mathbf{d}$ is not low for isomorphism.

If $\mathbf{d} \neq \mathbf{0}$ is low for isomorphism, then $\mathbf{d}$ is not a degree of categoricity because any computable structure which is $\mathbf{d}$-computable categorical is also computably categorical. However, the converse is not true because the degrees of categoricity are not closed upwards while the degrees which are not low for isomorphism are closed upwards. More specifically, Anderson and Csima show that every noncomputable hyperimmune-free degree is not a degree of categoricity, but it follows from the examples in Section 3 that there are hyperimmune-free degrees which are not low for isomorphism.

In Section 4, we consider the measure of the class of all sets which have low-for-isomorphism degree. Because this class is a Borel tailset, Kolmogorov's 0-1 Law implies that it must have measure 0 or 1. (See Barmpalias, Day and Lewis [2] for background on measure theoretic arguments in classical recursion theory.) We show that this class has measure 0 and that no Martin-Löf random degree can be low for isomorphism.

Finally, we will conclude with a brief discussion and some questions.

When working with the notion of lowness for isomorphism, it is convenient to work with computable structures in a fixed computable language rather than considering all computable structures across any computable language.

Proposition 1.3. *A degree* $\mathbf{d}$ *is low for isomorphism if and only if for every pair of computable directed graphs* G_0 *and* G_1, $G_0 \cong_{\mathbf{d}} G_1$ *if and only if* $G_0 \cong_{\Delta^0_1} G_1$.

Proof. Hirschfeldt, Khoussainov, Shore and Slinko [8] gave an effective method of coding an arbitrary countable structure $\mathcal{A}$ in a computable language into a countable directed graph $G(\mathcal{A})$ with the following properties.

- $\mathcal{A} \cong \mathcal{B}$ if and only if $G(\mathcal{A}) \cong G(\mathcal{A})$.
- $\mathcal{A}$ is computable if and only if $G(\mathcal{A})$ is computable.
- If $\mathcal{A}$ and $\mathcal{B}$ are computable, then for any Turing degree $\mathbf{d}$, $\mathcal{A} \cong_{\mathbf{d}} \mathcal{B}$ if and only if $G(\mathcal{A}) \cong_{\mathbf{d}} G(\mathcal{B})$.

The proposition follows immediately from this coding. $\square$

Hirschfeldt, Khoussainov, Shore and Slinko actually showed that there are several classes of universal structures in this sense and one could work with any of them in the context of low-for-isomorphism degrees. We choose directed graphs for convenience. However, this restriction to directed graphs raises the natural question of what happens if one restricts to a class of structures (such as linear orders, Boolean algebras or abelian groups) which are not universal in the sense described by Hirschfeldt, Khoussainov, Shore and Slinko [8].

Definition 1.4. Let $\mathcal{C}$ be a class of computable algebraic structures closed under isomorphism within the class of all computable structures. A degree $\mathbf{d}$ is *low for* $\mathcal{C}$*-isomorphism* if for every pairs of structures $\mathcal{A}, \mathcal{B} \in \mathcal{C}$, $\mathcal{A} \cong_{\mathbf{d}} \mathcal{B}$ if and only if $\mathcal{A} \cong_{\Delta^0_1} \mathcal{B}$.

In Section 5, we consider the low-for-$\mathcal{L}$-isomorphism degrees where $\mathcal{L}$ is the class of computable linear orders. We replicate the negative results from Section 3 within this class of degrees, but leave open the question of whether every low-for-$\mathcal{L}$-isomorphism degree is low for isomorphism.

2 Degrees which are low for isomorphism

In this section, we use Cohen, Mathias and Sacks forcing to construct low-for-isomorphism degrees. In the cases of Cohen and Mathias forcing, these facts follow immediately from known results about forcing for models of second order arithmetic. We assume familiarity with Cohen and Mathias forcing notions in recursion theory and refer the reader to Jockusch [11] and Cholak, Dzhafarov, Hirst and Slaman [3] for the relevant definitions. Given our applications, we sketch the background on models of second order arithmetic in the context of ω-models, although the results hold more generally. Simpson [16] contains a discussion of models of second order arithmetic and forcing in the more general context of these models.

$\mathcal{I} \subseteq \mathcal{P}(\omega)$ is a *Turing ideal* if $\mathcal{I}$ is closed under Turing reducibility and the Turing join. Given any set A, $\mathcal{I}_A = \{X \subseteq \omega \mid X \leq_T A\}$ is a Turing ideal and we refer to $\mathcal{I}_A$ as the ideal generated by A.

An ω-model $\mathcal{M}$ of RCA_0 consists of the standard model of PA (providing the range of the first order variables) together with a Turing ideal $\mathcal{I}_\mathcal{M}$ (providing the range of the second order variables). We will abuse notation by equating an ω-model $\mathcal{M}$ of RCA_0 with the corresponding ideal $\mathcal{I}_\mathcal{M}$. In particular, we say that $\mathcal{M}$ is countable if this ideal is countable, and we let $\mathcal{M}_A$ denote the countable ω-model given by the ideal $\mathcal{I}_A$.

If $\mathcal{M}$ is an ω-model of RCA_0 and G is a set, then $\mathcal{M}[G]$ denotes the smallest ω-model containing $\mathcal{M}$ and the set G. That is, the ideal corresponding to $\mathcal{M}[G]$ is the downward closure under $\leq_T$ of the set $\{X \oplus G \mid X \in \mathcal{M}\}$. In particular, $\mathcal{M}_A[G] = \mathcal{M}_{A\oplus G}$.

Let $\mathcal{M}$ be an ω-model of RCA_0. There is a Π^0_2 formula $\Phi_{\text{iso}}(X, Y)$ with two free set variables such that for any directed graphs $\mathcal{A}, \mathcal{B} \in \mathcal{M}$ and any function $f \in \mathcal{M}$, $\mathcal{M} \models \Phi_{\text{iso}}(\mathcal{A} \oplus \mathcal{B}, f)$ if and only if f is an isomorphism from $\mathcal{A}$ to $\mathcal{B}$.

We can now state the relevant facts concerning Cohen and Mathias forcing and apply these facts in our context.

Theorem 2.1. *Let $\mathcal{M}$ be a countable ω-model of RCA_0 and let $\Phi(X, Y)$ be a Σ^0_3 formula with two free set variables such that for some fixed $A \in \mathcal{M}$ there is no $B \in \mathcal{M}$ such that $\mathcal{M} \models \Phi(A, B)$.*

(I) (Hirschfeldt, Shore and Slaman [10]) If G is Cohen 2-generic over $\mathcal{M}$, then there is no $B \in \mathcal{M}[G]$ such that $\mathcal{M}[G] \models \Phi(A, B)$.

(II) (Hirschfeldt and Shore [9]) If G is Mathias 2-generic over $\mathcal{M}$, then there is no $B \in \mathcal{M}[G]$ such that $\mathcal{M}[G] \models \Phi(A, B)$.

Theorem 2.2. *Let $\mathcal{A}$ and $\mathcal{B}$ be computable directed graphs.*

(I) If U and V are sets such that V is Cohen 2-U-generic, then

$$\mathcal{A} \cong_{deg(U)} \mathcal{B} \Leftrightarrow \mathcal{A} \cong_{deg(U\oplus V)} \mathcal{B}.$$

(II) If U and V are sets such that V is Mathias 2-U-generic, then

$$\mathcal{A} \cong_{deg(U)} \mathcal{B} \Leftrightarrow \mathcal{A} \cong_{deg(U\oplus V)} \mathcal{B}.$$

Proof. The left-to-right implications are trivial. To prove the right-to-left directions, assume that $\mathcal{A} \not\cong_{\deg(U)} \mathcal{B}$. Consider the ω-model $\mathcal{M}_U$ of RCA_0 generated by U. Because $\mathcal{A} \not\cong_{\deg(U)} \mathcal{B}$, there is no $f \in \mathcal{M}$ such that $\mathcal{M}_U \models \Phi_{\text{iso}}(\mathcal{A} \oplus \mathcal{B}, f)$. Applying Theorem 2.1 (I) or (II), depending on the type of forcing, we conclude that there is no $f \in \mathcal{M}_U[V] = \mathcal{M}_{U\oplus V}$ such that $\mathcal{M}_{U\oplus V} \models \Phi_{\text{iso}}(\mathcal{A} \oplus \mathcal{B}, f)$. Hence $\mathcal{A} \not\cong_{\deg(U\oplus V)} \mathcal{B}$ as required. □

Corollary 2.3. *Every Cohen 2-generic degree and every Mathias 2-generic degree is low for isomorphism. Furthermore, if $\mathbf{d}$ is low for isomorphism, then there is a low-for-isomorphism degree $\mathbf{c} > \mathbf{d}$. Furthermore, for any n, we can ensure that $\mathbf{c}' \geq \mathbf{0^{(n)}}$.*

Proof. For the first statement, apply Theorem 2.2 with U computable and V any Cohen or Mathias 2-generic set. For the second statement, fix $D \in \mathbf{d}$, let V be Cohen (or Mathias) 2-D-generic and let $\mathbf{c} = \deg(D \oplus V)$. Since V is generic relative to D, $\mathbf{c} > \mathbf{d}$. Let $\mathcal{A}$ and $\mathcal{B}$ be computable directed graphs. Since $\mathbf{d}$ is low for isomorphism, $\mathcal{A} \cong_{\Delta^0_1} \mathcal{B}$ if and only if $\mathcal{A} \cong_{\mathbf{d}} \mathcal{B}$. By Theorem 2.2 (I) (or (II)), $\mathcal{A} \cong_{\mathbf{d}} \mathcal{B}$ if and only if $\mathcal{A} \cong_{\mathbf{c}} \mathcal{B}$. Hence, $\mathcal{A} \cong_{\Delta^0_1} \mathcal{B}$ if and only if $\mathcal{A} \cong_{\mathbf{c}} \mathcal{B}$ as required.

To show the last statement, we fix an n and $D \in \mathbf{d}$. We will define a sequence of sets $X_0 <_T X_1 <_T \ldots$ such that $\deg(X_n)$ is low for isomorphism and $X_n'' \leq_T X_{n+1}'$ for each n. It will follow that $0^{(n+1)} \leq_T X_n'$ for each n.

We first claim that if Y is 3-X-generic for X-computable Mathias forcing, then the principal function of Y, p_Y, dominates all functions computable in X. To see this, we fix an index e for which Φ_e^X is computable and a condition (F, C). We can X-computably thin C to a subset $C' \subseteq C$ such that the principal function $p_{F\cup C'}$ dominates Φ_e^X. Therefore, the set of conditions $(\widehat{F}, \widehat{C})$ for which $p_{\widehat{F}\cup\widehat{C}}$ dominates Φ_e^X is dense. Since this set of conditions is also Σ_3^X, every 3-X-generic set for X-computable Mathias forcing must meet each such set of conditions. Our claim follows because if $p_{\widehat{F}\cup\widehat{C}}$ dominates Φ_e^X and Y extends $(\widehat{F}, \widehat{C})$, then p_Y also dominates Φ_e^X.

Now the statement follows very quickly. Let $X_0 = D$ and assume that we have defined X_n and that $\deg(X_n)$ is low for isomorphism. Let Y_n be 3-X_n-generic for X_n-computable Mathias forcing. By Theorem 2.2, $\deg(X_n \oplus Y_n)$ is low for isomorphism as well. Now Martin's Theorem and our claim above show us that $(X_n \oplus Y_n)' \geq_T X_n''$, and we set $X_{n+1} = X_n \oplus Y_n$. □

From this corollary, we can infer the existence of low-for-isomorphism degrees with certain properties by appealing to the corresponding results for Cohen and Mathias n-generics for $n \geq 2$. For example, there are Δ^0_3 low-for-isomorphism degrees (since there are Δ^0_3 Cohen 2-generics), there are low-for-isomorphism degrees which are hyperimmune (since all Cohen 2-generic degrees are hyperimmune), there are low-for-isomorphism degrees which are not minimal (since no Cohen 2-generic degree is minimal) and there are low-for-isomorphism degrees in the jump classes $\mathbf{GL_1}$ and $\mathbf{GH_1}$ (since Cohen 2-generic degrees are in $\mathbf{GL_1}$ and Mathias 2-generic degrees are in $\mathbf{GH_1}$). We can even infer the existence of a high low-for-isomorphism degree. Furthermore, since the low-for-isomorphism degrees are closed downwards, there are low-for-isomorphism degrees in $\mathbf{GL_2}-\mathbf{GL_1}$ (because every Cohen 2-generic degree bounds a degree in this class) and in $\mathbf{GL_3}-\mathbf{GL_2}$ (because every Cohen 3-generic degree bounds a degree in this class).

We turn to Sacks forcing with computable perfect trees to obtain low-for-isomorphism degrees which are minimal and hyperimmune free. Although various people have observed that Sacks forcing can be used in this context, there does not appear to be a proof in the literature. We review the relevant definitions and lemmas, but refer the reader to Chapter V.5 in Odifreddi [15] for the proofs of the computation lemmas. We use λ to denote the empty string, $\alpha \sqsubseteq \beta$ to denote that the string α is an initial segment of β and $\alpha * \beta$ (or $\alpha * n$ if $\beta = \langle n \rangle$) to denote the concatenation of α and β.

Definition 2.4. Let $\alpha, \beta, \gamma \in 2^{<\omega}$. We say β and γ *split* α if $\alpha \sqsubseteq \beta$, $\alpha \sqsubseteq \gamma$ and β and γ are incomparable. We say β and γ *e-split* α if β and γ split α and there is an x such that

$\Phi_e^\beta(x)\downarrow \neq \Phi_e^\gamma(x)\downarrow$.

Definition 2.5. A *computable perfect tree* is a computable function $T : 2^{<\omega} \to 2^{<\omega}$ such that for all σ, $T(\sigma * 0)$ and $T(\sigma * 1)$ split $T(\sigma)$. We say that a string τ *is on* T if $\tau = T(\sigma)$ for some σ. We say that a set A *is on* T if for all n, there is an $m \geq n$ such that $A \upharpoonright m$ is on T.

Definition 2.6. Let T be a computable perfect tree. S is *a computable perfect subtree of* T if S is a computable perfect tree and for all σ, $S(\sigma) = T(\tau)$ for some string τ. For any string δ, the *full subtree of* T *above* δ is the subtree S defined by $S(\sigma) = T(\delta * \sigma)$ for all σ.

To construct a noncomputable hyperimmune-free degree by forcing with computable perfect trees, we use the following two standard lemmas.

Lemma 2.7. *For any computable perfect tree T and any index e, there is a computable perfect subtree S of T such that for all A on S, $A \neq \Phi_e$.*

Lemma 2.8. *For any computable perfect tree T and any index e, there is a computable perfect subtree S of T such that either Φ_e^A is not total for all A on S, or Φ_e^A is total for all A on S and $\Phi_e^{S(\sigma)}(n)\downarrow$ for all $n \leq |\sigma|$. In the latter case, for all A on S, Φ_e^A is majorized by the computable function $f(n) = \max\{\Phi_e^{S(\sigma)}(n) \mid |\sigma| = n\}$.*

Definition 2.9. A computable perfect tree is *e-splitting* if for all σ, $T(\sigma * 0)$ and $T(\sigma * 1)$ e-split $T(\sigma)$.

To construct a minimal degree by forcing with computable perfect trees, we use the following standard lemma.

Lemma 2.10. *For any computable perfect tree T and any index e, there is a computable perfect subtree S of T such that either*

- *for every A on S, if Φ_e^A is total, then Φ_e^A is computable, or*
- *for every A on S, if Φ_e^A is total, then $A \leq_T \Phi_e^A$.*

Theorem 2.11. *There is a degree* $\mathbf{d}$ *such that* $\mathbf{d}$ *is hyperimmune free, minimal and low for isomorphism.*

Proof. For this construction, we use forcing with perfect trees. We build a (noneffective) sequence of computable perfect trees

$$T_0 \supseteq T_1 \supseteq T_2 \supseteq \cdots$$

such that T_0 is the identity tree, T_{i+1} is a computable perfect subtree of T_i, and $T_i(\lambda) \subsetneq T_{i+1}(\lambda)$ for all i. We set D to be the unique set such that $T_i(\lambda) \sqsubseteq D$ for all i and let $\mathbf{d}$ be the degree of D. We will have four types of requirements in order to make $\mathbf{d}$ noncomputable, hyperimmune free, minimal and low for isomorphism. The first three parts are standard and we mention them only briefly.

To make **d** noncomputable, hyperimmune free and minimal, we meet the requirements

$$\mathrm{Diag}_e : D \neq \Phi_e,$$
$$\mathrm{HIFree}_e : \Phi_e^D \text{ is not total or } \Phi_e^D \text{ is majorized by a computable function, and}$$
$$\mathrm{Min}_e : \Phi_e^D \text{ total } \rightarrow (\Phi_e^D \text{ is computable or } D \leq_T \Phi_e^D)$$

for each e. Depending on which requirement has highest priority at stage $s+1$, we apply Lemma 2.7, 2.8 or 2.10 to T_s to obtain T_{s+1}.

We must now explain how to force **d** to be low for isomorphism. Fix a (noneffective) list $(\mathcal{A}_i, \mathcal{B}_i)$, $i \in \omega$, of all pairs of infinite computable directed graphs. (We assume without loss of generality that the domains of $\mathcal{A}_i$ and $\mathcal{B}_i$ are ω.) We meet the requirements

$$\mathrm{Low}_{\langle e,i\rangle} : \text{If } \Phi_e^D \text{ is an isomorphism } \mathcal{A}_i \rightarrow \mathcal{B}_i, \text{ then } \mathcal{A}_i \cong_{\Delta_1^0} \mathcal{B}_i.$$

Assume $\mathrm{Low}_{\langle e,i\rangle}$ is the highest priority requirement left at stage $s+1$. Without loss of generality, we assume that we satisfy the requirement HIFree_e before working on $\mathrm{Low}_{\langle e,i\rangle}$ for any i. If we satisfied HIFree_e by guaranteeing that Φ_e^D is not total, then $\mathrm{Low}_{\langle e,i\rangle}$ is also satisfied. Assume we satisfied HIFree_e by guaranteeing that Φ_e^D is total and majorized by a computable function. In this case, for all A on T_s, Φ_e^A is total. We proceed in two steps.

Step 1. Check (noneffectively) whether there is a string σ and a number n such that $\Phi_e^{T_s(\sigma)} \upharpoonright n \downarrow$ and $\Phi_e^{T_s(\sigma)} \upharpoonright n$ is not a partial isomorphism from $\mathcal{A}_i$ to $\mathcal{B}_i$. If there is such a string σ, then define T_{s+1} to be the full subtree of T_s above σ, skip Step 2 below and proceed to the next stage. In this case, for any A on T_{s+1} we have that $\Phi_e^A \upharpoonright n = \Phi_e^{T_s(\sigma)} \upharpoonright n$ and hence Φ_e^A is not an isomorphism from $\mathcal{A}_i$ to $\mathcal{B}_i$. In particular, we have satisfied $\mathrm{Low}_{\langle e,i\rangle}$. If there is no such string σ, proceed to Step 2.

Step 2. Check (noneffectively) whether there is a string σ and a number m such that for all strings $\tau \sqsupseteq \sigma$ and all numbers x, $\Phi_e^{T_s(\tau)}(x) \neq m$ (either by failing to converge or converging to a number other than m). If there is such a string σ, then define T_{s+1} to be the full subtree of T_s above σ. In this case, we have guaranteed that for each A on T_{s+1}, m is not in the range of Φ_e^A. Thus, we have satisfied $\mathrm{Low}_{\langle e,i\rangle}$ and can proceed to stage $s+1$.

If there is no such string σ and number m, then we want to define T_{s+1} to be a computable subtree of T_s such that for all A on T_{s+1}, Φ_e^A is onto. We define $T_{s+1}(\tau)$ by induction on the length of τ. Set $T_{s+1}(\lambda) = T_s(\langle 0\rangle)$. For the inductive case, assume $|\tau| = m$, $T_{s+1}(\tau)$ is defined and $T_{s+1}(\tau) = T_s(\delta)$. For $i \in \{0,1\}$, we computably search for a string σ_i and a number x such that $\delta * i \sqsubseteq \sigma_i$ and $T_e^{T_s(\sigma_i)}(x) = m$. By our case assumptions, this search must terminate. Set $T_{s+1}(\tau * i) = T_s(\sigma_i)$.

It remains to be shown that in this final case we have satisfied $\mathrm{Low}_{\langle e,i\rangle}$. Notice that T_{s+1} is a computable perfect tree with the following properties.

- For every A on T_{s+1}, Φ_e^A is total (by our action for HIFree_e).
- For every A on T_{s+1}, Φ_e^A is onto (by our failure to find a string σ in Step 2).
- For every string σ and number n, if $\Phi_e^{T_{s+1}(\sigma)} \upharpoonright (n+1) \downarrow$, then $\Phi_e^{T_{s+1}(\sigma)} \upharpoonright (n+1)$ is a partial isomorphism from $\mathcal{A}_i$ to $\mathcal{B}_i$ (by our failure to find a string σ in Step 1).

We claim that there is a computable isomorphism between $\mathcal{A}_i$ to $\mathcal{B}_i$, and hence we have satisfied $\mathrm{Low}_{\langle e,i\rangle}$.

The computable isomorphism is constructed by an effective back-and-forth argument. We define a computable sequence of finite binary strings $\sigma_0 \sqsubseteq \sigma_1 \sqsubseteq \sigma_2 \sqsubseteq \cdots$ and of numbers $n_0 < n_1 < \cdots$ such that $\Phi_e^{T_{s+1}(\sigma_i)} \upharpoonright (n_i + 1) \downarrow$. The partial isomorphism at stage i of the back-and-forth construction is $\Phi_e^{T_{s+1}(\sigma_i)} \upharpoonright (n_i + 1)$. To begin, set $\sigma_{-1} = \lambda$ and $n_{-1} = 0$.

Suppose we have constructed σ_j and the next back-and-forth action is to extend our partial isomorphism to include m in the domain. Let $n_{j+1} = \max\{m, n_j + 1\}$. Effectively search for a string τ such that $\sigma_j \sqsubseteq \tau$ and $\Phi_e^{T_{s+1}(\tau)} \upharpoonright (n_{j+1} + 1) \downarrow$. Since Φ_e^A is total for all A on T_{s+1}, this search must terminate. Moreover, the map given by $\Phi_e^{T_{s+1}(\tau)} \upharpoonright (n_{j+1} + 1)$ is a partial isomorphism which is consistent with our current partial isomorphism so we can set $\sigma_{j+1} = \tau$.

Similarly, if we have constructed σ_j and the next back-and-forth action is to extend our partial isomorphism to include m in the range, then effectively search for a string τ and a number $n > n_j$ such that $\sigma_j \sqsubseteq \tau$, $\Phi_e^{T_{s+1}(\tau)} \upharpoonright (n + 1) \downarrow$ and $\Phi_e^{T_{s+1}(\tau)}(k) = m$ for some $k \leq n$. Since Φ_e^A is total and onto for all A on T_{s+1}, this search must terminate. Again, the map given by $\Phi_e^{T_{s+1}(\tau)} \upharpoonright (n+1)$ is a partial isomorphism which is consistent with our current partial isomorphism, so we set $n_{j+1} = n$ and $\sigma_{j+1} = \tau$. This completes the proof. □

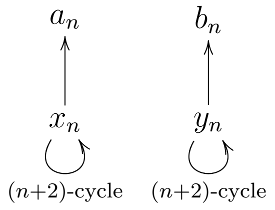
3 Degrees which are not low for isomorphism

Having constructed examples of degrees which are low for isomorphism and have various other properties, we turn to constructing examples which are not low for isomorphism. The theme connecting these results is that if $\mathbf{d}$ bounds a degree containing a set which can be nicely approximated in some sense, then it should be possible to use this approximation to diagonalize and build a pair of computable graphs which are not computable isomorphic but are $\mathbf{d}$-computably isomorphic.

Theorem 3.1. *If* $\mathbf{d}$ *is a noncomputable* Δ_2^0 *degree, then* $\mathbf{d}$ *is not low for isomorphism. Hence, no degree which bounds a noncomputable* Δ_2^0 *degree is low for isomorphism.*

Proof. Let D be a set of degree $\mathbf{d}$ and fix a Δ_2^0 approximation $\langle D_s \rangle$ to D. We assume that $D_0 = \emptyset$. We build a pair of computable directed graphs G and H such that there is a unique isomorphism $\alpha : G \to H$ and this isomorphism satisfies $\alpha \equiv_T D$.

We begin by placing a single $(n + 2)$-cycle in each of G and H for each n. Let x_n denote a fixed element of the $(n + 2)$-cycle in G and add an element a_n with an edge from x_n to a_n. Similarly, let y_n denote a fixed element of the $(n + 2)$-cycle in H and add an element b_n with an edge from y_n to b_n. For a fixed n, we refer to these components as the n-th components of G and H respectively. We can visualize these n-th components side by side as follows:



Throughout the construction, we maintain the property that there is a unique isomorphism between G and H and that this isomorphism matches up n-th components and sends x_n to y_n. When $n \notin D_s$, the isomorphism will send a_n to b_n, while if $n \in D_s$, the isomorphism will not map a_n to b_n. More specifically, the n-th components of G and H remain the same until the first stage s' (if any) at which $n \in D_{s'}$. At stage s', add new elements a' and b' (respectively) to the n-th component of G and H as follows.

$$\begin{tikzcd}
a_n \arrow[rr] & & a' & b' \arrow[rr] & & b_n \\
& x_n \arrow[ul] \arrow[ur] \arrow[loop below, "(n+2)\text{-cycle}"] & & & y_n \arrow[ul] \arrow[ur] \arrow[loop below, "(n+2)\text{-cycle}"] &
\end{tikzcd}$$

The unique isomorphism between G and H now sends a_n to b' and a' to b_n. We leave the n-th components unchanged until the next stage s'' (if any) at which $n \notin D_{s''}$. At stage s'', add new elements a'' and b'' to the n-th components as follows.

$$\begin{tikzcd}
a'' \arrow[r] & a_n \arrow[r] & a' & b' \arrow[r] & b_n \arrow[r] & b'' \\
& x_n \arrow[ul] \arrow[u] \arrow[ur] \arrow[loop below, "(n+2)\text{-cycle}"] & & & y_n \arrow[ul] \arrow[u] \arrow[ur] \arrow[loop below, "(n+2)\text{-cycle}"] &
\end{tikzcd}$$

We have restored the property that the unique isomorphism sends a_n to b_n. From here, the pattern repeats. Each time n enters D_s, we add a new element to each of the linear chains above x_n and y_n to ensure that a_n cannot map to b_n. When n leaves D_s, we add an element to each linear chain to ensure that a_n maps to b_n once again. Because D is Δ^0_2, such changes occur only finitely often for each n. Once D_s has stopped changing on n, the n-th components of G and H stabilize and the unique isomorphism maps a_n to b_n if and only if $n \notin D$. Therefore, for any degree $\mathbf{a}$, $G \cong_{\mathbf{a}} H$ if and only if $\mathbf{d} \leq \mathbf{a}$. $\square$

Corollary 3.2. *There are hyperimmune degrees, minimal degrees and* $\boldsymbol{GL}_1$ *degrees which are not low for isomorphism.*

Proof. These statements follow from Theorem 3.1 because every nonzero Δ^0_2 degree is hyperimmune, there are minimal Δ^0_2 degrees, and there are low Δ^0_2 degrees. $\square$

Corollary 3.3. *The low-for-isomorphism degrees are not closed under join.*

Proof. There are Cohen 2-generic degrees $\mathbf{a}$ and $\mathbf{b}$ such that $\mathbf{a} \cup \mathbf{b} \geq \mathbf{0}'$. $\square$

We can now observe that our result concerning Cohen 2-generics cannot be strengthened. It is clear that there are Cohen 1-generics that are not low for isomorphism since there are Δ^0_2 Cohen 1-generics. Furthermore, there are Cohen weak 2-generics that are not low for isomorphism because there are Cohen weak 2-generics above $\mathbf{0}'$.

Theorem 3.4. *Let X and Y be any pair of computably inseparable c.e. sets. No degree* $\mathbf{d}$ *which can compute a separating set for X and Y is low for isomorphism.*

Proof. The proof is similar to the proof of Theorem 3.1. Fix X and Y with their c.e. approximations $\langle X_s \rangle$ and $\langle Y_s \rangle$. We construct a pair of computable directed graphs G and H such that every isomorphism between G and H can compute a separating set for X and Y and such that every separating set for X and Y can compute an isomorphism.

At the start of the construction, for each $n \in \omega$, G and H each contain an $(n+2)$-cycle (called the n-th components). Let x_n (respectively y_n) denote a fixed element of the $(n+2)$-cycle in G (in H). Add two elements a_n and a'_n (respectively b_n and b'_n) to the n-th component of G (of H) with edges from x_n to a_n and a'_n (from y_n to b_n and b'_n). We can visualize these n-th components side by side as follows:

$$\begin{tikzcd}
a_n & & a'_n & b_n & & b'_n \\
& x_n \arrow[ul] \arrow[ur] \arrow[loop below, "(n+2)\text{-cycle}"] & & & y_n \arrow[ul] \arrow[ur] \arrow[loop below, "(n+2)\text{-cycle}"] &
\end{tikzcd}$$

As the construction proceeds, we maintain the property that any isomorphism between G and H must match up n-th components and send x_n to y_n. However, there may be more than one way to match up a_n and a'_n with b_n and b'_n and these choices are independent for each n. At the start, there are two options for each n; we can map a_n to b_n and a'_n to b'_n or map a_n to b'_n and a'_n to b_n. As long as $n \notin X_s \cup Y_s$, we continue to allow both options. However, if n enters X_s, we add elements a'' and b'' as follows:

$$\begin{tikzcd}
a_n & a'' \arrow[r] & a'_n & b_n & b'' \arrow[r] & b'_n \\
& x_n \arrow[ul] \arrow[u] \arrow[ur] \arrow[loop below, "(n+2)\text{-cycle}"] & & & y_n \arrow[ul] \arrow[u] \arrow[ur] \arrow[loop below, "(n+2)\text{-cycle}"] &
\end{tikzcd}$$

This forces the isomorphism to send a_n to b_n. On the other hand, if n enters Y_s, we add elements a'' and b'' as follows:

$$\begin{tikzcd}
a_n & a'' \arrow[r] & a'_n & b_n & b'' \arrow[l] & b'_n \\
& x_n \arrow[ul] \arrow[u] \arrow[ur] \arrow[loop below, "(n+2)\text{-cycle}"] & & & y_n \arrow[ul] \arrow[u] \arrow[ur] \arrow[loop below, "(n+2)\text{-cycle}"] &
\end{tikzcd}$$

This forces the isomorphism to send a_n to b'_n.

If $\alpha : G \to H$ is an isomorphism (at the end of the construction), then $\{n \mid \alpha(a_n) = b_n\}$ contains X and is disjoint from Y. Therefore, if $\mathbf{c}$ can compute an isomorphism, then $\mathbf{c}$ can compute a separating set, and hence $G \ncong_{\Delta^0_1} H$ since X and Y are computably inseparable.

On the other hand, suppose S is a separating set. We use S to define an isomorphism $\alpha : G \to H$. First, let $\alpha(x_n) = y_n$ and let α match the remainder of the $(n+2)$-cycles in G and H. If $n \in S$, set $\alpha(a_n) = b_n$ and $\alpha(a'_n) = b'_n$. If $n \notin S$, set $\alpha(a_n) = b'_n$ and $\alpha(a'_n) = b_n$. Note that if $n \notin X \cup Y$, then these conditions completely define α as an isomorphism between the n-th components. If $n \in X \cup Y$, then we define $\alpha(a'') = b''$ when the elements a'' and b'' enter G and H. Because $n \in X$ implies $n \in S$ and because $n \in Y$ implies $n \notin S$, these conditions determine an isomorphism between the n-th components as required. □

Corollary 3.5. *There are hyperimmune-free degrees which are not low for isomorphism.*

Proof. This corollary follows from the Hyperimmune-Free Basis Theorem for Π^0_1 classes. □

4 Measure

By the results of Sections 2 and 3, we know that the low-for-isomorphism degrees are large in the sense of category and that these degrees neither contain nor are disjoint from the minimal degrees, the hyperimmune-free degrees or the $\mathbf{GL}_1$ degrees. In this section, we show that the low-for-isomorphism degrees are small in the sense of measure and are disjoint from the Martin-Löf random degrees.

Theorem 4.1. *The set of degrees which are low for isomorphism has measure 0. Furthermore, no Martin-Löf random degree is low for isomorphism.*

The rest of this section is devoted to the proof of this theorem. First, we show that the set of degrees which are not low for isomorphism has measure 1. As noted in Section 1, it suffices to show that the set of degrees which are not low for isomorphism has measure at least 1/2. Second, we refine this construction to show that every Martin-Löf random degree is not low for isomorphism.

We build (classically) isomorphic computable directed graphs G and H and a Π^0_1 class $\mathcal{C} \subseteq 2^\omega$ with the following properties.

(P1) $G \not\cong_{\Delta^0_1} H$.

(P2) $\mu(\mathcal{C}) \geq 1/2$.

(P3) If $X \in \mathcal{C}$, then X computes an isomorphism from G to H.

Thus, $\mathcal{C}$ is a Π^0_1 class of positive measure such that the graphs G and H witness that every element of $\mathcal{C}$ is not low for isomorphism. To satisfy (P1), we meet the requirements

$$R_e : \Phi_e \text{ is not an isomorphism from } G \text{ to } H$$

for each e. To satisfy (P2), we ensure that the diagonalization strategy for R_e does not remove too much of the tree defining the Π^0_1 class $\mathcal{C}$. To satisfy (P3), we define a Turing functional Γ such that for all $X \in \mathcal{C}$, $\Gamma^X : G \to H$ is an isomorphism.

An *e-component* in G or H consists of an $(e+3)$-cycle with a coding node u distinguished by an edge $E(u,u)$. If $e \neq e'$, then an e-component is not isomorphic to an e'-component.

Furthermore, given two e-components, there is a unique isomorphism between them and this isomorphism matches the coding nodes.

A *tailed e-component* consists of an e-component together with two additional nodes x_0 and x_1 and edges $E(u, x_0)$, $E(x_0, x_1)$ and $E(x_1, x_0)$ where u is the coding node of the e-component. In other words, a tailed e-component is an e-component with a disjoint 2-cycle attached to the coding node. At any stage of the construction, we can convert an e-component into a tailed e-component by attaching a 2-cycle. We refer to this process as adding a tail to the respective e-component. At certain points, we may refer to an e-component as *untailed* to emphasize that it does not (yet) contain a tail.

The isomorphism type of G and H will consist of countably many untailed e-components for every e and, for each requirement R_e for which we actively diagonalize, countably many tailed e-components. For any set X, we will have $G \cong_X H$ if and only if X can compute a bijection between the coding nodes in G and the coding nodes in H which correctly matches the coding nodes of e-components with and without tails. That is, given such a bijection, X can effectively extend the bijection to a full isomorphism by matching up the elements in the corresponding cycles and in the corresponding tails.

We construct G and H in stages with G_s and H_s denoting these graphs at the end of stage s. At stage 0, G_0 and H_0 contain infinitely many untailed e-components for each e. We describe the intuition behind meeting a single R_e and defining $\mathcal{C}$ and Γ. We consider the interaction between these strategies for a single R_e and then give the full construction.

To meet a single R_e, we fix an e-component in G_0 and use its coding node a_e as a diagonalizing witness. If we never see a stage s at which $\Phi_{e,s}(a_e) = b$ for some coding node b of an e-component in H_s, then R_e is trivially satisfied. If $\Phi_{e,s}(a_e) = b$ for such a coding node b, then we actively diagonalize by adding tails to an infinite coinfinite set of e-components in H_s including the e-component coded by b. To maintain isomorphic structures, we also add tails to an infinite coinfinite set of e-components in G_s but we do not add a tail to the e-component coded by a_e in G_s. This action meets R_e, and we will not change the e-components in either structure after this stage. Note that all the components in G and H exist at stage 0 and the only change is to add tails to some of these components.

We define the Π^0_1 class $\mathcal{C}$ by building a computable sequence of trees $T_s \subseteq 2^{<\omega}$ such that $2^{<\omega} = T_0 \supseteq T_1 \supseteq \cdots$. We let $T = \cap_s T_s$ and define $\mathcal{C} = [T]$. (Recall that for any $\sigma \in 2^{<\omega}$, we define $[\sigma]$ to be the set of infinite binary strings X extending σ and for any subset S of $2^{<\omega}$, we define $[S]$ to be the set of infinite binary strings X extending some $\sigma \in S$.) At stage s, we say that we remove a string σ from T to mean that $\sigma \notin T_s$ and hence implicitly that $\tau \notin T_s$ for all τ extending σ. When removing σ from T at stage s, we do not assume that $\sigma \in T_{s-1}$. That is, we could have $\sigma \notin T_{s-1}$ because some initial segment of σ was removed from T at an earlier stage.

At stage 0, we define the Turing functional Γ so that for all $X \in [T_0]$, Γ^X is an isomorphism from G_0 to H_0. More specifically, for each e and each node $\sigma \in T_0$ with $|\sigma| = e + 2$, we define Γ^σ so that it matches up the coding nodes for e-components in G_0 in bijective correspondence with the coding nodes for e-components in H_0. For $\delta \neq \sigma$ with $|\delta| = e + 2$, the matching given by Γ^δ will not be the same as the matching given by Γ^σ.

These bijective matchings extend effectively to an isomorphism between G_0 and H_0. Because the only elements added at future stages are tails to some e-components in G_s or H_s, a bijective match between coding nodes for e-components in G and H will extend effectively to an isomorphism as long as it correctly matches the coding nodes for e-components with and without tails.

As the construction proceeds, we will need to deal with the following conflict. Suppose we see $\Phi_{e,s}(a_e) = b$ and add tails to some e-components to meet R_e. If we defined $\Gamma^\sigma(a_e) = b$ for some $\sigma \in T_{s-1}$ then the action for R_e will also diagonalize against Γ^X being an isomorphism for any X extending σ. Therefore, we need to remove σ from T, which will reduce the current measure of T by $2^{-|\sigma|}$. To make $\mu([T]) \geq 1/2$, we ensure that the strings removed from T for different R_e requirements are spread out enough to make the total measure removed sufficiently small. Specifically, each R_e will be allowed to remove at most $2^{-(e+2)}$ much measure.

We describe the full strategy for R_0 before giving the general construction. R_0 is allowed to remove at most $1/4$ measure from T, or in other words, it is allowed to remove at most one string at level two from T. Let σ_i for $i \leq 3$ denote the nodes at level 2 in $T_0 = 2^{<\omega}$.

At the beginning of the construction, we label the coding nodes for the 0-components in G_0 by a_0 (our distinguished diagonalizing witness) and $c_j^{\sigma_i}$ for $j \in \omega$ and $i \leq 3$. Similarly, we label the coding nodes for the 0-components in H_0 by $b_{0,i}$ for $i \leq 3$ and $d_j^{\sigma_i}$ for $j \in \omega$ and $i \leq 3$. We view these coding nodes in columns as follows.

G_0 coding nodes:

$$\begin{array}{ccccc}
a_0 & c_0^{\sigma_0} & c_0^{\sigma_1} & c_0^{\sigma_2} & c_0^{\sigma_3} \\
 & c_1^{\sigma_0} & c_1^{\sigma_1} & c_1^{\sigma_2} & c_1^{\sigma_3} \\
 & c_2^{\sigma_0} & c_2^{\sigma_1} & c_2^{\sigma_2} & c_2^{\sigma_3} \\
 & c_3^{\sigma_0} & c_3^{\sigma_1} & c_3^{\sigma_2} & c_3^{\sigma_3} \\
 & c_4^{\sigma_0} & c_4^{\sigma_1} & c_4^{\sigma_2} & c_4^{\sigma_3} \\
 & \vdots & \vdots & \vdots & \vdots
\end{array}$$

H_0 coding nodes:

$$\begin{array}{ccccc}
b_{0,0} & d_0^{\sigma_0} & d_0^{\sigma_1} & d_0^{\sigma_2} & d_0^{\sigma_3} \\
b_{0,1} & d_1^{\sigma_0} & d_1^{\sigma_1} & d_1^{\sigma_2} & d_1^{\sigma_3} \\
b_{0,2} & d_2^{\sigma_0} & d_2^{\sigma_1} & d_2^{\sigma_2} & d_2^{\sigma_3} \\
b_{0,3} & d_3^{\sigma_0} & d_3^{\sigma_1} & d_3^{\sigma_2} & d_3^{\sigma_3} \\
 & d_4^{\sigma_0} & d_4^{\sigma_1} & d_4^{\sigma_2} & d_4^{\sigma_3} \\
 & \vdots & \vdots & \vdots & \vdots
\end{array}$$

For each $i \leq 3$, we define Γ^{σ_i} to give a bijection between these coding nodes. First, for each infinite column other than the i-th column, we match the coding nodes in order.

$$\text{For } \ell \neq i, \text{ set } \Gamma^{\sigma_i}(c_k^{\sigma_\ell}) = d_k^{\sigma_\ell} \text{ for all } k \in \omega.$$

Second, we match a_0 with $b_{0,i}$ by setting $\Gamma^{\sigma_i}(a_0) = b_{0,i}$. Third, we use $c_0^{\sigma_i}$, $c_1^{\sigma_i}$ and $c_2^{\sigma_i}$ to match with $\{b_{0,0}, b_{0,1}, b_{0,2}, b_{0,3}\} \setminus \{b_{0,i}\}$ in order of the indices and then match the remainder of the i-th infinite column in G with the i-th infinite column in H by shifting the indices.

$$\Gamma^{\sigma_i}(c_k^{\sigma_i}) = \begin{cases} b_{0,k} & \text{if } k < i \\ b_{0,k+1} & \text{if } i \leq k < 3 \\ d_{k-3}^{\sigma_i} & \text{if } 3 \leq k \end{cases}$$

To give one full picture, here is the bijection given by Γ^{σ_1}.

$$\begin{array}{cccc}
c_0^{\sigma_0} \mapsto d_0^{\sigma_0} & c_0^{\sigma_1} \mapsto b_{0,0} & c_0^{\sigma_2} \mapsto d_0^{\sigma_2} & c_0^{\sigma_3} = d_0^{\sigma_3} \\
c_1^{\sigma_0} \mapsto d_1^{\sigma_0} & a_0 \mapsto b_{0,1} & c_1^{\sigma_2} \mapsto d_1^{\sigma_2} & c_1^{\sigma_3} = d_1^{\sigma_3} \\
\vdots & c_1^{\sigma_1} \mapsto b_{0,2} & \vdots & \vdots \\
 & c_2^{\sigma_1} \mapsto b_{0,3} & & \\
 & c_3^{\sigma_1} \mapsto d_0^{\sigma_1} & & \\
 & c_4^{\sigma_1} \mapsto d_1^{\sigma_1} & & \\
 & \vdots & &
\end{array}$$

As the construction proceeds, we wait for a stage s such that $\Phi_{0,s}(a_0) = b$ for some coding node b in a 0-component in H_s. There are two possible cases we need to consider at such a stage, and we will act differently in each case.

For the first case, suppose that $\Phi_{e,s}(a_0) = d_j^{\sigma_\ell}$ for some $\ell \leq 3$ and $j \in \omega$. For each $i \neq \ell$, we have defined $\Gamma^{\sigma_i}(c_k^{\sigma_\ell}) = d_k^{\sigma_\ell}$ for all $k \in \omega$. Therefore, without disrupting Γ^{σ_i} for $i \neq \ell$, we can add tails to all 0-components in G_s with coding nodes of the form $c_k^{\sigma_\ell}$ and to all 0-components in H_s with coding nodes of the form $d_k^{\sigma_\ell}$. Because we add a tail to $d_j^{\sigma_\ell}$ but not to a_0, we meet R_e. However, because Γ^{σ_ℓ} matches up some coding nodes of the form $c_k^{\sigma_\ell}$ with coding nodes of the form $b_{0,k'}$ (where $k = k'$ or $k + 1 = k'$), the bijection given by Γ^{σ_ℓ} no longer correctly matches coding nodes with and without tails. Therefore, we remove σ_ℓ from T and we have permanently won R_e at the expense of only $1/4$ measure.

For the second case, suppose that $\Phi_{0,s}(a_0) = b_{0,i}$ for some $i \leq 3$. In this case, we add a tail to $b_{0,i}$ in H_s but do not add a tail to a_0 in G_s. This action wins R_e but because $\Gamma^{\sigma_i}(a_0) = b_{0,i}$, the bijection given by Γ^{σ_i} is no longer correct and we must remove σ_i from T losing $1/4$ measure. We need to add tails to other 0-components in G_s and H_s to ensure that each of the other bijections Γ^{σ_j} for $j \neq i$ continues to be a bijection.

Fix such a $j \leq 3$. Consider the values of Γ^{σ_j} on the j-th infinite column of coding nodes $c_k^{\sigma_j}$. Γ^{σ_j} either maps $c_i^{\sigma_j}$ to $b_{0,i}$ (if $j > i$) or maps $c_{i-1}^{\sigma_j}$ to $b_{0,i}$ (if $j < i$). Fix k_j such that $\Gamma^{\sigma_j}(c_{k_j}^{\sigma_j}) = b_{0,i}$. Since we add a tail to $b_{0,i}$, we must add a tail to $c_{k_j}^{\sigma_j}$ for Γ^{σ_j} to remain correct.

Now consider an arbitrary index $n \leq 3$ with $n \neq i, j$. Because $n \neq j$, we have $\Gamma^{\sigma_n}(c_k^{\sigma_j}) = d_k^{\sigma_j}$ for all k. In particular, $\Gamma^{\sigma_n}(c_{k_j}^{\sigma_j}) = d_{k_j}^{\sigma_j}$ so we need to add a tail to $d_{k_j}^{\sigma_j}$ for Γ^{σ_n} to remain correct. However, we have already defined $\Gamma^{\sigma_j}(c_{k_j+3}^{\sigma_j}) = d_{k_j}^{\sigma_j}$ so we have to add a tail to $c_{k_j+3}^{\sigma_j}$ for Γ^{σ_j} to remain correct. This pattern repeats. $\Gamma^{\sigma_n}(c_{k_j+3}^{\sigma_j}) = d_{k_j+3}^{\sigma_j}$ so we add a tail to $d_{k_j+3}^{\sigma_j}$ for the sake of Γ^{σ_n}. Then, $\Gamma^{\sigma_j}(c_{k_j+6}^{\sigma_j}) = d_{k_j+3}^{\sigma_j}$ forces us to add a tail to $c_{k_j+6}^{\sigma_j}$ for the sake of Γ^{σ_j} and so on.

Therefore, for each $j \leq 3$ with $j \neq i$, we add tails to all coding nodes of the form $c_{k_j+3m}^{\sigma_j}$ and $d_{k_j+3m}^{\sigma_j}$ for $m \in \omega$. This action respects the definition of Γ^{σ_j} because $\Gamma^{\sigma_j}(c_{k_j}^{\sigma_j}) = b_{0,i}$ and $\Gamma^{\sigma_j}(c_{k_j+3(m+1)}^{\sigma_j}) = d_{k_j+3m}^{\sigma_j}$. It also respects Γ^{σ_n} for $n \neq i, j$ because $\Gamma^{\sigma_n}(c_{k_j+3m}^{\sigma_j}) = d_{k_j+3m}^{\sigma_j}$. Thus, we can add tails to an infinite and coinfinite set of coding nodes in G_s and H_s in a manner that preserves the bijections given by Γ^{σ_j} for $j \neq i$ and that wins R_e at the cost of removing a single node σ_i with $|\sigma_i| = 2$ from T.

This completes the description of satisfying a single requirement R_0 while building T. The general construction proceeds by using nodes at level $e + 2$ of T to meet R_e. The strategy for R_e will remove at most one node from T at level $e+2$. In particular, there is no injury between R_e strategies for different indices e. We now give the full construction.

At stage 0, we set up the full construction as follows. For each fixed $e \in \omega$, let $\langle \sigma_{e,i} \rangle_{i<2^{e+2}}$ be a list of the binary strings of length $e+2$. G_0 will have infinitely many e-components with the coding nodes denoted by a_e and $c_j^{\sigma_{e,i}}$ for $j \in \omega$ and $i < 2^{e+2}$, and H_0 will have infinitely many e-components with the coding nodes denoted by $b_{e,i}$ for $i < 2^{e+2}$ and $d_j^{\sigma_{e,i}}$ for $j \in \omega$ and $i < 2^{e+2}$. For each $i < 2^{e+2}$, we make the following definitions for $\Gamma^{\sigma_{e,i}}$.

$$\Gamma^{\sigma_{e,i}}(a_e) = b_{e,i} \text{ and } \Gamma^{\sigma_{e,i}}(c_k^{\sigma_{e,\ell}}) = d_k^{\sigma_{e,\ell}} \text{ for } \ell \neq i \text{ and } k \in \omega, \text{ and}$$

$$\Gamma^{\sigma_{e,i}}(c_k^{\sigma_{e,i}}) = \begin{cases} b_{e,k} & \text{if } k < i \\ b_{e,k+1} & \text{if } i \leq k < 2^{e+2}-1 \\ d_{k-(2^{e+2}-1)}^{\sigma_{e,i}} & \text{if } 2^{e+2}-1 \leq k \end{cases}$$

At stage $s > 0$, we actively diagonalize for each R_e such that $\Phi_{e,s}(a_e)$ converges to a coding node for an e-component in H_{s-1} and we have not yet actively diagonalized for R_e. The action we take depends on the output of $\Phi_{e,s}(a_e)$.

Case 1. Suppose that $\Phi_e(a_e) = d_j^{\sigma_{e,i}}$ for some $j \in \omega$ and $i < 2^{e+2}$. Fix the value i.

- Remove $\sigma_{e,i}$ from T.
- Add a tail to each e-component in G_s with coding node $c_\ell^{\sigma_{e,i}}$ for $\ell \in \omega$.
- Add a tail to each e-component in H_s with coding node $d_\ell^{\sigma_{e,i}}$ for $\ell \in \omega$.

Case 2. Suppose that $\Phi_e(a_e) = b_{e,i}$ for some $i < 2^{e+2}$. Fix the value i.

- Remove $\sigma_{e,i}$ from T.
- Add a tail to the e-component in H_s with coding node $b_{e,i}$.
- For each $j < 2^{e+2}$ with $j \neq i$, let k_j be such that $\Gamma^{\sigma_{e,j}}(c_{k_j}^{\sigma_{e,j}}) = b_{e,i}$.
  - Add a tail to each e-component in G_s with coding node $c_{k_j+\ell\cdot(2^{e+2}-1)}^{\sigma_{e,j}}$ for $\ell \in \omega$.
  - Add a tail to each e-component in H_s with coding node $d_{k_j+\ell\cdot(2^{e+2}-1)}^{\sigma_{e,j}}$ for $\ell \in \omega$.

Lemma 4.2. *$G \not\cong_{\Delta_1^0} H$.*

Proof. We need to show that each R_e is satisfied. If we never see a stage s such that $\Phi_{e,s}(a_e) = b$ for a coding node b of an e-component in H_{s-1}, then R_e is satisfied because a_e is a coding node for an e-component and $\Phi_e(a_e)$ is not.

If we do see such a stage, fix the least s at which $\Phi_{e,s}(a_e) = b$ for a coding node b of an e-component in H_{s-1}. In both cases of the construction at stage s, the e-component of H_s with coding node b is given a tail but the e-component of G_s with coding node a_e is not given a tail. Since R_e has diagonalized at stage s, it never acts again and hence every tailed e-component of G receives its tail at stage s. Therefore, a_e remains the coding location of an untailed e-component in G while $\Phi_e(a_e)$ becomes the coding location of a tailed e-component in H and hence R_e is satisfied. □

Lemma 4.3. *The measure $\mu([T])$ is at least $1/2$.*

Proof. Each requirement R_e acts at most once and removes a single node $\sigma_{e,i}$ from T when it acts. Since $|\sigma_{e,i}| = 2^{e+2}$, the total measure removed from $[T]$ is bounded by $\sum_{e \in \omega} 2^{-e-2} = 1/2$. Therefore, $\mu([T]) \geq 1 - 1/2 = 1/2$. □

Lemma 4.4. *For each $X \in [T]$, Γ^X is a bijection between the coding nodes in G and H which correctly matches e-components with and without tails in these graphs.*

Proof. Fix e. We claim that for every s and $X \in [T_s]$, Γ^X is a bijection between the coding nodes for e-components in G_s and H_s which correctly matches the coding nodes with and without tails. Because an e-component which has a tail in G or H receives this tail at some finite stage, this claim suffices to establish the lemma.

We prove the claim by induction on s. When $s = 0$, fix an arbitrary set X (since $[T_0] = 2^\omega$) and index e. Fix the index n such that $X \upharpoonright e + 2 = \sigma_{e,n}$. The definitions for $\Gamma^{\sigma_{e,n}}$ given at stage 0 bijectively match the coding nodes for e-components in G_0 with the coding nodes for e-components in H_0. Since no components are tailed at stage 0, this bijection correctly matches those components with and without tails.

For the inductive case, assume the condition in the claim holds at stage $s - 1$. Fix a set $X \in [T_s]$ and an index e. We split into two cases. For the first case, assume that R_e does not act at stage s. In this case, no tails are added to e-components at stage s. Since $X \in [T_s] \subseteq [T_{s-1}]$, the induction hypothesis implies that Γ^X correctly matches the coding nodes for e-components with and without tails in G_{s-1} and H_{s-1}. Since no e-components receive a tail at stage s, this matching remains correct at stage s.

For the second case, assume that R_e acts at stage s. Fix the index n such that $X \upharpoonright e + 2 = \sigma_{e,n}$. We split into subcases depending on whether R_e acts in Case 1 or Case 2 of the construction. Suppose that $\Phi_e(a_e) = d_j^{\sigma_{e,i}}$ so R_e acts in Case 1 of the construction. Because $X \in [T_s]$, we know that $\sigma_{e,n}$ was not removed from T at stage s and hence $n \neq i$. The e-components which receive tails in G_s and H_s are exactly those with coding nodes $c_\ell^{\sigma_{e,i}}$ and $d_\ell^{\sigma_{e,i}}$. However, since $n \neq i$, we have $\Gamma^{\sigma_{e,n}}(c_\ell^{\sigma_{e,i}}) = d_\ell^{\sigma_{e,i}}$ for all ℓ. Hence Γ^X correctly matches up the coding nodes which receive tails at stage s.

For the final subcase, assume that $\Phi_e(a_e) = b_{e,i}$ so R_e acts in Case 2 of the construction. Since $X \in [T_s]$, we know $n \neq i$. For each $j < 2^{e+2}$ with $j \neq i$, fix k_j as in Case 2. An e-component receives a tail in G_s if and only if its coding node is $c_{k_j+\ell(2^{e+2}-1)}^{\sigma_{e,j}}$ for some $j \neq i$ and $\ell \in \omega$. It receives a tail in H_s if and only if its coding node is $b_{e,i}$ or $d_{k_j+\ell(2^{e+2}-1)}^{\sigma_{e,j}}$ for some $j \neq i$ and $\ell \in \omega$. If $j \neq n$, then $\Gamma^{\sigma_{e,n}}(c_{k_j+\ell(2^{e+2}-1)}^{\sigma_{e,j}}) = d_{k_j+\ell(2^{e+2}-1)}^{\sigma_{e,j}}$. If $j = n$, then $\Gamma^{\sigma_{e,n}}(c_{k_n}^{\sigma_{e,n}}) = b_{e,i}$ and $\Gamma^{\sigma_{e,n}}(c_{k+(2^{e+2}-1)}^{\sigma_{e,n}}) = d_k^{\sigma_{e,n}}$. Therefore, $\Gamma^{\sigma_{e,n}}(c_{k_n+(\ell+1)(2^{e+2}-1)}^{\sigma_{e,n}}) = d_{k_n+\ell(2^{e+2}-1)}^{\sigma_{e,n}}$ and the bijection of coding nodes for e-components given by $\Gamma^{\sigma_{e,n}}$ correctly matches up those receiving tails at stage s. □

This completes the proof that the measure of the degrees which are not low for isomorphism is 1. In the remainder of this section, we show that this result can be strengthened to show that no Martin-Löf random degree is low for isomorphism, and hence no degree which can compute a Martin-Löf random is low for isomorphism. We recall the definition of Martin-Löf randomness here (for a general reference, see [5] or [13]):

Definition 4.5. A Martin-Löf test is an effectively c.e. sequence $\langle V_i \rangle$ of subsets of $2^{<\omega}$ such that $\mu[V_i] \leq 2^{-i}$ for all i. A real X is Martin-Löf random if for every Martin-Löf test, $X \notin \cap_i [V_i]$.

The general construction of our Π^0_1 class above is quite flexible. For example, we obtained $\mu(\mathcal{C}) \geq 1/2$ by assigning each R_e the nodes at level $e+2$. By spreading the requirements out more sparsely in the tree, we could increase the measure of $\mathcal{C}$. To build a sequence of Π^0_1 classes that can be used to define the Σ^0_1 classes that compose an Martin-Löf test, we use a different type of flexibility.

Recall that λ denotes the empty sequence. As before, for a sequence $\tau \in 2^{<\omega}$, we let $[\tau]$ denote the set of all X such that $\tau \sqsubseteq X$. For a set X, we say that a triple (G, H, Γ) *witnesses that X is not low for isomorphism* if G and H are computable graphs such that $G \ncong_{\Delta^0_1} H$ and Γ is a Turing functional such that Γ^X is an isomorphism between G and H. For example, in the general construction, the triple (G, H, Γ) consisting of the graphs and the Turing functional built during the construction witnesses that each $X \in \mathcal{C}$ is not low for isomorphism.

Think of the general construction above as building a Π^0_1 class $\mathcal{C}$ contained in the clopen set $[\lambda]$. More generally, we can define a $[\tau]$-construction for any τ as follows. Run the construction above in $[\tau]$ by starting with $T_0 = [\tau]$ and assigning R_e the nodes $\sigma_{e,i}$ such that $\tau \sqsubseteq \sigma_{e,i}$ and $|\sigma_{e,i}| = |\tau| + e + 2$. Let G_τ and H_τ denote the computable graphs constructed, let S_τ denote the computable tree obtained at the end of the construction with these starting conditions (i.e. S_τ is the tree denoted by T in the general construction), let $\mathcal{C}_\tau = [S_\tau]$ denote the associated Π^0_1 class and let Γ_τ denote the Turing functional built in this altered construction. By the proofs given in the general construction, we have that $\mu(\mathcal{C}_\tau) \geq 2^{-|\tau|-1}$ (we lose at most half the measure in $[\tau]$), $G_\tau \ncong_{\Delta^0_1} H_\tau$ (because we meet the R_e requirements) and for every $X \in [S_\tau] = \mathcal{C}_\tau$, Γ^X_τ is an isomorphism between G_τ and H_τ. Therefore, the triple $(G_\tau, H_\tau, \Gamma_\tau)$ witnesses that each $X \in \mathcal{C}_\tau$ is not low for isomorphism.

To show that no Martin-Löf random degree is low for isomorphism, we construct an effective sequence of nested Π^0_1 classes $\mathcal{B}_0 \subseteq \mathcal{B}_1 \subseteq \cdots$ given by an effective sequence of nested trees $B_0 \subseteq B_1 \subseteq \cdots$ such that for every n, $\mu(\mathcal{B}_n) \geq 1 - 2^{-n+1}$ and for every $X \in \mathcal{B}_n$, X is not low for isomorphism. If A is Martin-Löf random, then $A \in \mathcal{B}_n$ for some n and hence A is not low for isomorphism.

We construct $\mathcal{B}_n = [B_n]$ by induction on n. Let $B_0 = T$ and $\mathcal{B}_0 = \mathcal{C} = [T]$ from the general construction. That is, we build B_0 by a $[\lambda]$-construction, setting $B_0 = S_\lambda$ and $\mathcal{B}_0 = [S_\lambda]$. The triple $(G_\lambda, H_\lambda, \Gamma_\lambda)$ witnesses that each $X \in \mathcal{B}_0$ is not low for isomorphism.

To build B_1, we run the $[\lambda]$-construction to build B_0 except rather than removing $\sigma_{e,i}$ from S_λ when we actively diagonalize, we start a $[\sigma_{e,i}]$-construction inside the clopen set $[\sigma_{e,i}]$ which runs concurrently with the $[\lambda]$-construction. More formally, let I_0 be the set of pairs $\langle e, i \rangle$ such that during the construction of B_0, we actively diagonalize to meet R_e by removing $\sigma_{e,i}$ from B_0. Then,

$$B_1 = B_0 \cup \bigcup_{\langle e,i \rangle \in I_0} S_{\sigma_{e,i}}.$$

Equivalently, the complement of B_1 is generated by the set of strings δ which are removed during a $[\sigma_{e,i}]$-construction initiated by an active diagonalization process for B_0. The set of strings generating the complement of B_1 is c.e., so $\mathcal{B}_1 = [B_1]$ is a Π^0_1 class containing $\mathcal{B}_0$.

Because $\mu([S_{\sigma_{e,i}}]) \geq 2^{-|\sigma_{e,i}|-1}$ and $|\sigma_{e,i}| = e+2$, we have $\mu([S_{\sigma_{e,i}}]) \geq 2^{-e-3}$. Since the $[\sigma_{e,i}]$-construction works inside $[\sigma_{e,i}]$ and $\mu([\sigma_{e,i}]|) = 2^{-e-2}$, the $[\sigma_{e,i}]$-construction removes at most 2^{-e-3} much measure from $[\sigma_{e,i}]$ during its construction. For each e, there is at most one i such that $\langle e,i\rangle \in I_0$. Therefore,

$$\mu(\mathcal{B}_1) \geq 1 - \sum_{e\in\omega} 2^{-e-3} = 1 - 1/4 = 3/4$$

as required.

Furthermore, if $X \in \mathcal{B}_1$, then either $X \in \mathcal{B}_0$ or $X \in [S_{\sigma_{e,i}}]$ for some $\langle e,i\rangle \in I_0$. If $X \in \mathcal{B}_0$, then $(G_\lambda, H_\lambda, \Gamma_\lambda)$ witnesses that X is not low for isomorphism. If $X \in [S_{\sigma_{e,i}}]$ for $\langle e,i\rangle \in I_0$, then $(G_{\sigma_{e,i}}, H_{\sigma_{e,i}}, \Gamma_{\sigma_{e,i}})$ witnesses that X is not low for isomorphism. This completes the verification that $\mathcal{B}_1$ has the required properties.

The general induction strategy follows the same pattern in a nested fashion. Given B_n, we define I_n to be the set of pairs $\langle e,i\rangle$ such that during the construction of B_n, we actively diagonalize to meet R_e by removing $\sigma_{e,i}$ from B_n. Note that we only put $\langle e,i\rangle$ in I_n if the construction of B_n actually removes $\sigma_{e,i}$ as opposed to starting a $[\sigma_{e,i}]$-construction inside $[\sigma_{e,i}]$ as directed by the inductive construction of B_n.

To build B_{n+1}, we run the construction of B_n except rather than removing from $\sigma_{e,i}$ from B_n when $\langle e,i\rangle \in I_n$, we start a $[\sigma_{e,i}]$-construction inside $[\sigma_{e,i}]$. Thus $B_{n+1} = B_n \cup \bigcup_{\langle e,i\rangle\in I_n} S_{\sigma_{e,i}}$. Equivalently, the complement of B_{n+1} is generated by the strings δ which are removed during a $[\sigma_{e,i}]$-construction initialized by an active diagonalization process for B_n which actually removed $\sigma_{e,i}$ from B_n, i.e. when $\langle e,i\rangle \in I_n$. Thus, $\mathcal{B}_{n+1} = [B_{n+1}]$ is a Π^0_1 class containing $\mathcal{B}_n$.

By induction, the construction of B_n removes at most 2^{-n+1} measure, so

$$\sum_{\langle e,i\rangle\in I_n} \mu([\sigma_{e,i}]) = \sum_{\langle e,i\rangle\in I_n} 2^{-e-2} \leq 2^{-n+1}.$$

In the construction of B_{n+1}, rather than removing all of $[\sigma_{e,i}]$ when $\langle e,i\rangle \in I_n$, we start a $[\sigma_{e,i}]$-construction which removes at most $2^{-|\sigma_{e,i}|-1}$ much measure. Therefore, in B_{n+1}, the measure removed is bounded above by

$$\sum_{\langle e,i\rangle\in I_n} 2^{-e-3} \leq 2^{-1}\cdot 2^{-n+1} = 2^{-(n+1)+1}$$

so $\mu(\mathcal{B}_{n+1}) \geq 1 - 2^{-(n+1)+1}$ as required.

Finally, if $X \in \mathcal{B}_{n+1}$, then either $X \in \mathcal{B}_n$ or $X \in [S_{\sigma_{e,i}}]$ for some $\langle e,i\rangle \in I_n$. If $X \in \mathcal{B}_n$, then X is not low for isomorphism by induction. If $X \in [S_{\sigma_{e,i}}]$, then $(G_{\sigma_{e,i}}, H_{\sigma_{e,i}}, \Gamma_{\sigma_{e,i}})$ witnesses that X is not low for isomorphism. This completes the verification that $\mathcal{B}_{n+1}$ satisfies the required properties and the proof that every Martin-Löf random is not low for isomorphism.

We note that this result cannot be strengthened in an obvious way. While no Martin-Löf random is low for isomorphism, we can observe that there are computable randoms that are low for isomorphism: every high degree contains a computably random real [14], and we know that there is a high degree that is low for isomorphism by Corollary 2.3.

5 Low for linear order isomorphism

In this section, we consider the notion of low for isomorphism restricted to a class of computable structures which are not computationally universal in the sense of Proposition 1.3. We reproduce the analog of Theorem 3.4 for computable linear orders. The same techniques suffice to prove the analog of Theorem 3.1 for linear orders, but, rather than give this proof, we describe a stronger result which will appear in Suggs [18].

Definition 5.1. A degree $\mathbf{d}$ is *low for LO-isomorphism* if for every pair of computable linear orders L_0 and L_1, $L_0 \cong_{\mathbf{d}} L_1$ if and only if $L_0 \cong_{\Delta^0_1} L_1$.

Theorem 5.2. *Let X and Y be any pair of computably inseparable c.e. sets. No degree $\mathbf{d}$ which can compute a separating set for X and Y is low for LO-isomorphism.*

Proof. Fix X and Y. It suffices to build isomorphic computable linear orders L_0 and L_1 such that every separating set for X and Y can compute an isomorphism from these linear orders and every isomorphism between them can compute a separating set for X and Y.

Let $\mathbb{Q}$ denote the countable dense linear order without endpoints and let $\mathbb{Z}$ denote the order type of the integers. L_0 and L_1 will be computably decomposable as

$$\begin{array}{ll} L_0 : & \mathbb{Q} + A_0 + \mathbb{Q} + A_1 + \mathbb{Q} + \cdots + \mathbb{Q} + A_n + \mathbb{Q} + \cdots \\ L_1 : & \mathbb{Q} + B_0 + \mathbb{Q} + B_1 + \mathbb{Q} + \cdots + \mathbb{Q} + B_n + \mathbb{Q} + \cdots \end{array}$$

where for each n, $A_n \cong B_n$ will either be isomorphic to $\mathbb{Z}$ or to a finite linear order. Thus any isomorphism between these linear orders has to match up each pair $A_n \cong B_n$ and has to match up the corresponding padding copies of $\mathbb{Q}$. Because L_0 and L_1 computably decompose into these forms, we know which points lie in each A_n or B_n component and which points lie in each $\mathbb{Q}$ component. Thus, to compute an isomorphism, it suffices to uniformly compute isomorphisms between A_n and B_n for each n. Conversely, every isomorphism computes a uniform sequence of isomorphisms between A_n and B_n.

We build the sequence of orders A_n and B_n in stages as follows. At stage 0, A_n and B_n are each a sequence of three points

$$\begin{array}{ll} A_n : & \alpha_n^{-1} < a_n < \alpha_n^1 \\ B_n : & \beta_n^{-1} < b_n < \beta_n^1. \end{array}$$

At each stage $s > 0$, we proceed in one of three cases. First, if $n \notin X_s \cup Y_s$, then we add a new pairs of points to each of A_n and B_n on the outside of the existing points. In this case, at the end of stage s, we have

$$\begin{array}{ll} A_n : & \alpha_n^{-(s+1)} < \alpha_n^{-s} < \cdots < \alpha_n^{-1} < a_n < \alpha_n^1 < \cdots < \alpha_n^s < \alpha_n^{s+1} \\ B_n : & \beta_n^{-(s+1)} < \beta_n^{-s} < \cdots < \beta_n^{-1} < b_n < \beta_n^1 < \cdots < \beta_n^s < \beta_n^{s+1}. \end{array}$$

Note that if $n \notin X \cup Y$, then A_n and B_n grow into copies of $\mathbb{Z}$ and it is possible for an isomorphism between A_n and B_n to map a_n to b_n or to map a_n to an element other than b_n.

Second, if s is the first stage at which $n \in X_s$, then we do not add any points to A_n or B_n at stage s or at any future stages. Therefore, the final forms of A_n and B_n are

$$\begin{array}{ll} A_n : & \alpha_n^{-s} < \cdots < \alpha_n^{-1} < a_n < \alpha_n^1 < \cdots < \alpha_n^s \\ B_n : & \beta_n^{-s} < \cdots < \beta_n^{-1} < b_n < \beta_n^1 < \cdots < \beta_n^s. \end{array}$$

In this case, A_n and B_n have finished growing and the unique isomorphism between them maps a_n to b_n.

Third, if s is the first stage at which $n \in Y_s$, then we add one point on the right end of A_n and one point on the left end of B_n. We do not add any further points to A_n or B_n at future stages. Therefore, the final forms of A_n and B_n are

$$\begin{array}{ll} A_n : & \alpha_n^{-s} < \cdots < \alpha_n^{-1} < a_n < \alpha_n^1 < \cdots < \alpha_n^s < \alpha_n^{s+1} \\ B_n : & \beta_n^{-(s+1)} < \beta_n^{-s} < \cdots < \beta_n^{-1} < b_n < \beta_n^1 < \cdots < \beta_n^s. \end{array}$$

In this case, A_n and B_n have finished growing and the unique isomorphism between them maps a_n to β_n^{-1}, and hence not to b_n.

This completes the description of the construction of L_0 and L_1. By construction, if $f : L_0 \to L_1$ is an isomorphism, then $\{n \mid f(a_n) = b_n\}$ is a separating set for X and Y. Therefore, each isomorphism computes a separating set.

Conversely, if U is a separating set, namely $X \subseteq U$ and $Y \cap U = \emptyset$, then we can build an isomorphism computably in U by mapping $a_n \mapsto b_n$ for all $n \in U$ and $a_n \mapsto \beta_n^{-1}$ for all $n \notin U$. Since the successor and predecessor relations are uniformly computable in each A_n and B_n (by construction), we can effectively extend this map across each pair A_n and B_n, and since $\mathbb{Q}$ is computably categorical, we can extend the map across each padding $\mathbb{Q}$ component. Therefore, each separating set computes an isomorphism. □

A similar proof using $\mathbb{Q}$ padding blocks can be given for the following theorem.

Theorem 5.3. *If* $\mathbf{d}$ *is a noncomputable* Δ^0_2 *degree, then* $\mathbf{d}$ *is not low for LO-isomorphism.*

Rather than give a proof of Theorem 5.3, we state a stronger forthcoming result.

Definition 5.4. A degree $\mathbf{d}$ is *low for ω-isomorphism* if for every pair L_0, L_1 of computable copies of the linear order ω, $L_0 \cong_{\mathbf{d}} L_1$ if and only if $L_0 \cong_{\Delta^0_1} L_1$.

Theorem 5.5 (Suggs [18]). *If* $\mathbf{d}$ *is a noncomputable* Δ^0_2 *degree, then* $\mathbf{d}$ *is not low for ω-isomorphism.*

Corollary 5.6 (Suggs [18]). *A degree* $\mathbf{d}$ *is not low for ω-isomorphism if and only if* $\mathbf{d}$ *bounds a noncomputable* Δ^0_2 *degree.*

6 Conclusion and questions

We have discussed the relationship of lowness for isomorphism to other properties that demonstrate some sort of computational weakness such as minimality, hyperimmune-freeness, and we have found that there is no clear relationship between lowness for isomorphism and any of the properties mentioned: there are low-for-isomorphism degrees that possess each of these properties and low-for-isomorphism degrees that don't. In addition, we have found natural classes of degrees that are low for isomorphism (for instance, the Cohen 2-generics) and natural classes of degrees that are not (for instance, the Martin-Löf randoms). We observe that in each of these cases, our bound is tight: there is a weak 2-generic that is not low for isomorphism, while there is a computably random degree that is. However, we do not have a full classification of the degrees that are low for isomorphism. Just as all of the known examples of degrees of categoricity contain sets that are computably approximable in some way, it seems that the nontrivial degrees that are low for isomorphism resist computable approximability in a very strong way.

Question 6.1. Characterize the degrees that are low for isomorphism.

Perhaps the following approach to this question might be useful. Since the set of degrees of categoricity is countable and no Martin-Löf random degree is low for isomorphism, we can see that almost every degree (in terms of measure) is neither. It may make sense to try to identify classes of degrees that fall into neither of these categories.

Question 6.2. Is there a natural class of degrees that contains no degrees of categoricity and no low-for-isomorphism degrees?

One possibility may be the degrees that are Cohen 1-generic but not Cohen 2-generic: these are not generic enough to guarantee lowness for isomorphism, and they are not so computably approximable that they seem likely to be degrees of categoricity.

References

[1] Bernard A. Anderson and Barbara F. Csima, "Degrees that are not degrees of categoricity," to appear.

[2] George Barmpalias, Adam Day and Andrew Lewis, "Typicality in the Turing Degrees," to appear.

[3] Peter A. Cholak, Damir D. Dzhafarov, Jeffry L. Hirst and Theodore A. Slaman, "Generics for computable Mathias forcing," to appear.

[4] Barbara F. Csima, Johanna N.Y. Franklin and Richard A. Shore, "Degrees of categoricity and the hyperarithmetic hierarchy," *Notre Dame Journal of Formal Logic* **54** (2013), 215-231.

[5] Rodney G. Downey and Denis R. Hirschfeldt, *Algorithmic Randomness and Complexity*, Springer–Verlag, Heidelberg, 2010.

[6] Ekaterina B. Fokina, Iskander Kalimullin and Russell Miller, “Degrees of categoricity of computable structures,” *Archive for Mathematical Logic* **49** (2010), 51-67.

[7] Johanna N.Y. Franklin, “Lowness and highness properties for randomness notions,” in *Proceedings of the 10th Asian Logic Conference* ed. T. Arai et al., World Scientific, 2010, 124-151.

[8] Denis R. Hirschfeldt, Bakhadyr Khoussainov, Richard A. Shore and Arkadii M. Slinko, “Degree spectra and computable dimensions in algebraic structures,” *Annals of Pure and Applied Logic* **115** (2002), 71-113.

[9] Denis R. Hirschfeldt and Richard A. Shore, “Combinatorial principles weaker than Ramey’s Theorem for Pairs,” *Journal of Symbolic Logic* **72** (2007), 171-206.

[10] Denis R. Hirschfeldt, Richard A. Shore and Theodore A. Slaman, “The Atomic Model Theorem and type omitting,” *Transactions of the American Mathematical Society* **361** (2009), 5805-5837.

[11] Carl G. Jockusch, Jr., “Degrees of generic sets,” in *Recursion theory: its generalizations and applications* ed. F.R. Drake and S.S. Wainer, Cambridge University Press, 1980, 110-139.

[12] Manuel Lerman, *Degrees of unsolvability*, Springer–Verlag, Heidelberg, 1983.

[13] André Nies, *Computability and randomness*, Oxford University Press, Oxford, 2009.

[14] André Nies, Frank Stephan and Sebastiaan A. Terwijn, “Randomness, relativization and Turing degrees,” *Journal of Symbolic Logic* **70** (2005), 515-535.

[15] Piergiorgio Odifreddi, *Classical recursion theory*, North-Holland, Amsterdam, 1989.

[16] Stephen G. Simpson, *Subsystems of second order arithmetic*, Springer–Verlag, Heidelberg, 1999.

[17] R.I. Soare, *Recursively enumerable sets and degrees*, Springer–Verlag, Heidelberg, 1987.

[18] J. Suggs, PhD thesis, University of Connecticut, in preparation.