COMPUTABLE CATEGORICITY RELATIVE TO A C.E. DEGREE

JAVA DARLEEN VILLANO

ABSTRACT. A computable graph $\mathcal{G}$ is computably categorical relative to a degree $\mathbf{d}$ if and only if for all $\mathbf{d}$-computable copies $\mathcal{B}$ of $\mathcal{G}$, there is a $\mathbf{d}$-computable isomorphism $f : \mathcal{G} \to \mathcal{B}$. In this paper, we prove that for every computable partially ordered set P and computable partition $P = P_0 \sqcup P_1$, there exists a computable computably categorical graph $\mathcal{G}$ and an embedding h of P into the c.e. degrees where $\mathcal{G}$ is computably categorical relative to all degrees in $h(P_0)$ and not computably categorical relative to any degree in $h(P_1)$. This is a generalization of a result by Downey, Harrison-Trainor, and Melnikov in [DHTM21].

1. Introduction

In computable structure theory, we are interested in effectivizing model theoretic notions and constructions. For a general background on computable structure theory, see Ash and Knight [Ash00] or Montalbán [Mon21]. In particular, many people have examined the complexity of isomorphisms between structures within the same isomorphism type. We restrict ourselves to countable structures in a computable language and assume their domain is ω.

A computable structure $\mathcal{A}$ is **computably categorical** if for any computable copy $\mathcal{B}$ of $\mathcal{A}$, there exists a computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$. There are many known examples of computably categorical structures including computable linear orderings with only finitely many adjacent pairs [Rem81], computable fields of finite transcendence degree [Erš77], and computable ordered groups of finite rank [GLS03]. In each of these examples, the condition given turns out to be both necessary and sufficient for computable categoricity. There are

Date: April 18, 2025.

2020 *Mathematics Subject Classification.* 03C57, 03D25.

Key words and phrases. Computable structure theory, computable categoricity, relative categoricity, computably enumerable degrees.

also examples of computable structures which are not computably categorical, such as $(\mathbb{N}, \leq)$ as a linear order.

We are also interested in studying relativizations of computable categoricity. The most studied relativization of this notion is relative computable categoricity. A computable structure $\mathcal{A}$ is **relatively computably categorical** if for any copy $\mathcal{B}$ of $\mathcal{A}$, there exists a $\mathcal{B}$-computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$. We can think of this relativization as relativizing categoricity to *all* degrees at once since we do not fix the complexity of our copies of $\mathcal{A}$. If a structure $\mathcal{A}$ is relatively computably categorical, then it is computably categorical. The converse does not hold in general, but often holds for structures where there is a purely algebraic characterization of computable categoricity. In particular, the examples of computable categorical structures listed above are also relatively computably categorical.

The connection between a purely algebraic characterization of computable categoricity and the equivalence of computable categoricity and relative computable categoricity was clarified by the following result which was independently discovered by Ash, Knight, Manasse, and Slaman [Ash+89], and Chisholm [Chi90].

Theorem 1.1 (Ash et al., Chisholm)**.** *A structure $\mathcal{A}$ is relatively computably categorical if and only if it has a formally Σ^0_1 Scott family.*

In this paper, we study the following relativization of categoricity, where for a computable structure $\mathcal{A}$, we consider noncomputable copies of $\mathcal{A}$ and ask for isomorphisms between $\mathcal{A}$ and those copies to have the same complexity as the copies.

Definition 1.2. Let $X \in 2^{\mathbb{N}}$. A structure $\mathcal{A}$ is **computably categorical relative to X** if and only if for all X-computable copies $\mathcal{B}$ of $\mathcal{A}$, there is an X-computable isomorphism $f : \mathcal{A} \to \mathcal{B}$.

We have that a computable structure $\mathcal{A}$ is relatively computably categorical when it is computably categorical relative to all $X \in 2^{\mathbb{N}}$. Given a computable structure $\mathcal{A}$, we can consider the set of $X \in 2^{\mathbb{N}}$ such that $\mathcal{A}$ is computably categorical relative to X (or not). By Martin's result in [Mar68], we obtain that either this set contains a cone in the Turing degrees or is disjoint from one. This motivates us to include the following definition which appears in a paper by Csima and Harrison-Trainor [CHT17].

Definition 1.3. A structure $\mathcal{A}$ is **computably categorical on a cone above d** if for all $\mathbf{c} \geq \mathbf{d}$, whenever $\mathcal{B}$ and $\mathcal{C}$ are $\mathbf{c}$-computable copies of $\mathcal{A}$, there is a $\mathbf{c}$-computable isomorphism between $\mathcal{B}$ and $\mathcal{C}$.

We can think of $\mathcal{A}$ being relatively computably categorical as it being computably categorical on a cone above $\mathbf{0}$. We also have that a structure $\mathcal{A}$ is computably categorical on a cone above $\mathbf{d}$ if and only if it is computably categorical relative to $\mathbf{c}$ for every $\mathbf{c} \geq \mathbf{d}$. Downey, Harrison-Trainor, and Melnikov in [DHTM21] showed that for every computable structure $\mathcal{A}$, the behavior of being categorical relative to a degree stabilizes on a cone above $\mathbf{0}''$. That is, either $\mathcal{A}$ is computably categorical relative to all degrees above $\mathbf{0}''$, or to no degree above $\mathbf{0}''$. Moreover, if a computable structure $\mathcal{A}$ is computably categorical on a cone above any degree $\mathbf{d}$, then it is also computably categorical on a cone above $\mathbf{d} \oplus \mathbf{0}''$. Since $\mathbf{d} \oplus \mathbf{0}'' \geq \mathbf{0}''$, by the previous fact, we get that $\mathcal{A}$ is computably categorical on a cone above $\mathbf{0}''$. Therefore, by Martin's result, the set of degrees $\mathbf{d}$ such that $\mathcal{A}$ is computably categorical relative to $\mathbf{d}$ either contains the cone above $\mathbf{0}''$ or does not contain any cone at all.

The situation below $\mathbf{0}'$ was shown to be vastly different by the authors in [DHTM21] proving that being computably categorical relative to a degree is not a monotonic property of a structure in the following way.

Theorem 1.4 (Downey, Harrison-Trainor, Melnikov). *There is a computable structure $\mathcal{A}$ and c.e. degrees $0 = Y_0 <_T X_0 <_T Y_1 <_T X_1 <_T \dots$ such that*

1. *$\mathcal{A}$ is computably categorical relative to Y_i for each i,*
2. *$\mathcal{A}$ is not computably categorical relative to X_i for each i,*
3. *$\mathcal{A}$ is relatively computably categorical to $\mathbf{0}'$.*

In this paper, we extend this result to partial orders of c.e. degrees, exhibiting that for the particular structure we build, the full extent in which it can change its categorical behavior relative to some c.e. degrees.

Theorem 1.5. *Let $P = (P, \leq)$ be a computable partially ordered set and let $P = P_0 \sqcup P_1$ be a computable partition. Then, there exists a computable computably categorical directed graph $\mathcal{G}$ and an embedding h of P into the c.e. degrees where $\mathcal{G}$ is computably categorical relative to each degree in $h(P_0)$ and is not computably categorical relative to each degree in $h(P_1)$.*

The proof is a priority construction on a tree of strategies, using several key ideas from the proof of Theorem 1.4 in [DHTM21] along with some new techniques. In Section 2, we introduce informal descriptions for the strategies we need to satisfy our requirements for the construction and discuss important interactions between certain strategies. In Section 3, we detail the formal strategies, state and prove auxiliary lemmas about our construction, and state and prove the main verification lemma.

1.1. **Acknowledgements.** I would like to thank my adviser Reed Solomon for suggesting this project and providing many helpful comments as I wrote this paper. I would also like to thank the anonymous referee for their valuable comments and suggestions for improvement.

This research was partially supported by a Focused Research Group grant from the National Science Foundation of the United States, DMS-1854355.

2. Informal strategies

To prove Theorem 1.5, we have four goals to achieve within our construction, giving us four types of requirements to satisfy. In this section, we give informal descriptions of the strategies needed to satisfy each requirement in isolation, and then describe the interactions which arise when we employ these strategies together.

2.1. **Embedding P into the c.e. degrees.** We embed the poset P into the c.e. degrees in a standard way by constructing an independent family of uniformly c.e. sets A_p for $p \in P$. We fix the following notation:

$$\widehat{D_p} := \bigoplus_{q \neq p} A_q.$$

For each $p \in P$, we ensure that $A_p \not\leq_T \widehat{D_p}$. The image of p will be the c.e. set $D_p = \bigoplus_{q \leq p} A_q$. Because the A_p are independent, our embedding is order-preserving, i.e., $p \leq q$ in P if and only if $D_p \leq_T D_q$.

For each $p \in P$ and $e \in \omega$, we define the **independence requirement**:

$$N_e^p : \Phi_e^{\widehat{D_p}} \neq A_p.$$

In order to satisfy an N_e^p requirement in isolation, we use the following N_e^p-strategy. Let α be an N_e^p-strategy. When α is first eligible to act, it picks a large number x_α. Once x_α is defined, α checks if $\Phi_e^{\widehat{D_p}}(x_\alpha)[s] \downarrow= 0$. If not, α takes no action at stage s. If $\Phi_e^{\widehat{D_p}}(x_\alpha)[s] \downarrow= 0$, then α enumerates x_α into A_p and preserves this computation by restraining $\widehat{D_p} \upharpoonright (\text{use}(\Phi_e^{\widehat{D_p}}(x_\alpha)) + 1)$.

Notice that if we never see that $\Phi_e^{\widehat{D_p}}(x_\alpha) \downarrow= 0$, then either $\Phi_e^{\widehat{D_p}}(x_\alpha) \uparrow$ or $\Phi_e^{\widehat{D_p}}(x_\alpha) \downarrow\neq 0$, and in either case, the value of $\Phi_e^{\widehat{D_p}}(x_\alpha)$ will not be equal to $A_p(x_\alpha) = 0$ and so we meet the N_e^p requirement. Otherwise, at the first stage s for which $\Phi_e^{\widehat{D_p}}(x_\alpha)[s] \downarrow= 0$, we enumerate x_α into A_p and restrain $\widehat{D_p}$ below $\text{use}(\Phi_e^{\widehat{D_p}}(x_\alpha)) + 1$. In this case we have that $\Phi_e^{\widehat{D_p}}(x_\alpha) \downarrow= 0 \neq 1 = A_p(x_\alpha)$, and so we satisfy N_e^p.

2.2. Making $\mathcal{G}$ computably categorical.

We will build $\mathcal{G}$ in stages. At stage $s = 0$, we set $\mathcal{G} = \emptyset$. Then, at stage $s > 0$, we add two new connected components to $\mathcal{G}[s]$ by adding the root nodes a_{2s} and a_{2s+1} for those components, and attaching to each node a 2-loop (a cycle of length 2). We then attach a $(5s + 1)$-loop to a_{2s} and a $(5s + 2)$-loop to a_{2s+1}. This gives us the configuration of loops:

$$a_{2s} : 2, 5s + 1$$

$$a_{2s+1} : 2, 5s + 2.$$

The connected component consisting of the root node a_{2s} with its attached loops will be referred to as the **2sth connected component** of $\mathcal{G}$. During the construction, we might add more loops to connected components of $\mathcal{G}$, which causes them to have the following configuration:

$$a_{2s} : 2, 5s+1, 5s+3$$

$$a_{2s+1} : 2, 5s+1, 5s+4$$

We might also later add more loops to this configuration to obtain the following configuration:

$$a_{2s} : 2, 5s+1, 5s+2, 5s+3$$

$$a_{2s+1} : 2, 5s+1, 5s+2, 5s+4.$$

The idea behind adding these loops is to uniquely identify each connected component of $\mathcal{G}$. In all configurations above, there is only one way to match the components in $\mathcal{G}$ with components in a computable graph in order to define an embedding.

To make $\mathcal{G}$ computably categorical, we attempt to build an embedding of $\mathcal{G}$ into each computable directed graph. For each index e, let $\mathcal{M}_e$ be the (partial) computable graph with domain ω such that $E(x,y) \iff \Phi_e(x,y) = 1$ and $\neg E(x,y) \iff \Phi_e(x,y) = 0$. If Φ_e is not total, then $\mathcal{M}_e$ will not be a computable graph, but we will attempt to embed $\mathcal{G}$ into $\mathcal{M}_e$ anyway since we cannot know whether Φ_e is total or not. So we have the following requirement for each $e \in \omega$.

$$S_e : \text{if } \mathcal{G} \cong \mathcal{M}_e, \text{ then there exists a computable isomorphism } f_e : \mathcal{G} \to \mathcal{M}_e$$

To satisfy each S_e requirement in isolation, we have the following strategy. Let α be an S_e-strategy. When α is first eligible to act, it sets its parameter $n_\alpha = 0$ and defines its current map f_α from $\mathcal{G}$ into $\mathcal{M}_e$ to be empty. For the rest of this description, let $n = n_\alpha$. This parameter will keep track of the connected components that α is attempting to match between $\mathcal{G}$ and $\mathcal{M}_e$, and will be incremented by 1 only when we find copies of the $2n$th and $(2n+1)$st connected components of $\mathcal{G}$. Suppose the map $f_\alpha[s-1]$ matches up the $2m$th and $(2m+1)$st components of $\mathcal{G}[s-1]$ and $\mathcal{M}_e[s-1]$ for all $m < n$. At future stages, α checks whether $\mathcal{M}_e[s]$ contains isomorphic copies of the $2n$th and $(2n+1)$st components in $\mathcal{G}[s]$.

If not, α takes no additional action at stage s and retains the parameter n and the map f_α. If $\mathcal{M}_e[s]$ contains copies of these components, then α extends the map by matching those components in $\mathcal{G}[s]$ and $\mathcal{M}_e[s]$, and it increments the value of n by 1. Since $\mathcal{G}$ and $\mathcal{M}_e$ are computable graphs, f_α will be correct on these components if $\mathcal{G} \cong \mathcal{M}_e$. When this isolated S_e-strategy is enacted along with the other strategies for other types of requirements, it may be the case that the initial definition of f_α on a pair of components is not total on all cycles attached to each root node, but if $\mathcal{G} \cong \mathcal{M}_e$, we will have a systematic way of extending f_α on any new cycles that were added (see Section 2.5).

If α finds copies of the $2n$th and $(2n+1)$st components of $\mathcal{G}$ for every n, then f_α will be a computable embedding of $\mathcal{G}$ into $\mathcal{M}_e$. Because of the form of $\mathcal{G}$, if $\mathcal{M}_e \cong \mathcal{G}$, then f_α will be a partial embedding which can be extended computably to a computable embedding on all of $\mathcal{G}$, satisfying the S_e requirement. Otherwise, there exists some n such that the $2n$th and $(2n+1)$st components of $\mathcal{G}$ were never matched, and so $\mathcal{G}$ and $\mathcal{M}_e$ cannot be isomorphic and so we trivially satisfy S_e.

2.3. **Being computably categorical relative to a degree.** In this construction, we want to define computations using a D_p-oracle that can be destroyed later by enumerating numbers into A_p. We achieve this by setting the use of the D_p-computation to be $\langle u, p\rangle$. Enumerating u into A_p causes $\langle u, p\rangle$ to enter D_p, destroying the associated computation.

For each $p \in P_0$, we ensure $\mathcal{G}$ is computably categorical relative to D_p. Let $\mathcal{M}_i^{D_p}$ be the (partial) D_p-computable directed graph with domain ω and edge relation given by $\Phi_i^{D_p}$. Since each D_p is c.e., we define the following terms to keep track of certain finite subgraphs which appear and remain throughout our construction.

Definition 2.1. Let C_0 and C_1 be isomorphic finite distinct subgraphs of $\mathcal{M}_i^{D_p}[s]$. The **age of $\mathbf{C_0}$** is the least stage $t \leq s$ such that all edges in C_0 appear in $\mathcal{M}_i^{D_p}[t]$, denoted by $\text{age}(C_0)$. We say that $\mathbf{C_0}$ is **older than $\mathbf{C_1}$** when $\text{age}(C_0) \leq \text{age}(C_1)$.

We say that C_0 is the **oldest** if for all finite distinct subgraphs $C \cong C_0$ of $\mathcal{M}_i^{D_p}[s]$, $\text{age}(C_0) \leq \text{age}(C)$.

Definition 2.2. Let $C_0 = \langle a_0, a_1, \ldots, a_k\rangle$ and $C_1 = \langle b_0, b_1, \ldots, b_k\rangle$ be isomorphic finite distinct subgraphs of $\mathcal{M}_i^{D_p}[s]$ with $a_0 < a_1 < \cdots < a_k$ and $b_0 < b_1 < \cdots < b_k$. We say that $C_0 <_{\text{lex}} C_1$ if for the least j such that $a_j \neq b_j$, $a_j < b_j$.

We say that C_0 is the **lexicographically least** if for all finite distinct subgraphs $C \cong C_0$ of $\mathcal{M}_i^{D_p}[s]$, $C_0 <_{\text{lex}} C$.

If $\mathcal{G} \cong \mathcal{M}_i^{D_p}$, then we need to build a D_p-computable isomorphism between these graphs. To achieve this, we meet the following requirement for each $i \in \omega$.

$$T_i^p : \text{if } \mathcal{G} \cong \mathcal{M}_i^{D_p}, \text{ then there exists a } D_p\text{-computable isomorphism } g_i^{D_p} : \mathcal{G} \to \mathcal{M}_i^{D_p}$$

The strategy to satisfy each T_i^p requirement in isolation is similar to the S_e-strategy, with some additional changes. Since the graphs are D_p-computable, embeddings defined by a T_i^p-strategy may become undefined later when small numbers enter D_p. Enumerations into D_p also cause changes in $\mathcal{M}_i^{D_p}$ such as disappearing edges. We will show in the verification that if $\mathcal{G} \cong \mathcal{M}_i^{D_p}$, "true" copies of components from $\mathcal{G}$ will eventually appear and remain in $\mathcal{M}_i^{D_p}$ (and thus become the oldest finite subgraph which is isomorphic to a component in $\mathcal{G}$), and so our T_i^p-strategy below will be able to define the correct D_p-computable isomorphism between the two graphs.

Let α be a T_i^p-strategy. When α is first eligible to act, it sets its parameter $n_\alpha = 0$ and defines $g_\alpha^{D_p}$ to be the empty map. Once α has defined n_α, then at the previous stage $s-1$ (or the last α-stage in the full construction), we have the following situation:

- For each $m < n_\alpha$, $g_\alpha^{D_p}[s-1]$ maps the $2m$th and $(2m+1)$st components of $\mathcal{G}[s-1]$ to isomorphic copies in $\mathcal{M}_i^{D_p}[s-1]$.
- For $m < n_\alpha$, let l_m be the maximum $\Phi_i^{D_p}[s-1]$-use for the loops in the copies in $\mathcal{M}_i^{D_p}[s-1]$ for the $2m$th and $(2m+1)$st components in $\mathcal{G}$. We can assume that if $m_0 < m_1 < n_\alpha$, then $l_{m_0} < l_{m_1}$.
- For $m < n_\alpha$, let $\langle u_m, p\rangle$ be the $g_\alpha^{D_p}[s-1]$-use for the mapping of the $2m$th and $(2m+1)$st components of $\mathcal{G}$. This use will be constant for all elements in these components.

- By construction, we will have that $l_m < u_k \leq \langle u_k, p\rangle$ for all $m \leq k < n_\alpha$.

Suppose α is acting at stage s and has already defined n_α. We first check whether numbers have been enumerated into D_p that injure the loops in $\mathcal{M}_i^{D_p}[s-1]$ given by $\Phi_i^{D_p}[s-1]$; that is, if there exists a number $x < \mathrm{use}(\Phi_i^{D_p}[s-1])$ which entered D_p. If not, then we keep the current value of n_α and skip ahead to the next step. If so, then let k be the least such that some number $x \leq l_k$ was enumerated into D_p. The loops in $\mathcal{M}_i^{D_p}[s]$ in the copies of the $2k$th and $(2k+1)$st components of $\mathcal{G}$ have been injured, and so may have disappeared. We have that $x \leq \langle u_m, p\rangle$ for all $k \leq m < n_\alpha$, and so our map $g_\alpha^{D_p}[s]$ is now undefined on all the $2m$th and $(2m+1)$st components for $k \leq m < n_\alpha$. So, α redefines $n_\alpha = k$ to find new images for the $2k$th and $(2k+1)$st components in $\mathcal{G}[s]$.

Second, we check to see if there is a $j < k$ and a new $x \in D_p$ such that $l_j < x < \langle u_j, p\rangle$. By minimality of the value k above, we know that $l_j < x$ for all $j < k$ and $x \in D_p[s] \setminus D_p[s-1]$. For each such j, our map $g_\alpha^{D_p}[s-1]$ has been injured on the $2j$th and $(2j+1)$st components of $\mathcal{G}$, but the loops in the copies of those components in $\mathcal{M}_i^{D_p}[s]$ remain intact. Therefore, we define $g_\alpha^{D_p}[s]$ on these components with oracle $D_p[s]$ to be equal to $g_\alpha^{D_p}[s-1]$. Furthermore, we keep the same use for $g_\alpha^{D_p}[s]$ on these components. This will ensure that injury of this type happens only finitely often.

Third, we check whether we can extend $g_\alpha^{D_p}[s-1]$ to the $2n_\alpha$th and $(2n_\alpha+1)$st components of $\mathcal{G}[s]$. Search for isomorphic copies in $\mathcal{M}_i^{D_p}[s]$ of these components. If there are multiple copies in $\mathcal{M}_i^{D_p}[s]$, choose the oldest such copy to map to, and if there are multiple equally old copies, choose the lexicographically least oldest copy. If there are no copies in $\mathcal{M}_i^{D_p}[s]$, then keep the value of n_α the same and $g_\alpha^{D_p}$ unchanged and let the next requirement act. Otherwise, extend $g_\alpha^{D_p}[s-1]$ to $g_\alpha^{D_p}[s]$ to include the $2n_\alpha$th and $(2n_\alpha+1)$st components of $\mathcal{G}$ and set the use to be $\langle u_{n_\alpha}, p\rangle$ where u_{n_α} is large (and so $u_{n_\alpha} > l_k$ for all $k \leq n_\alpha$). Increment n_α by 1 and go to the next requirement.

If $\mathcal{G} \cong \mathcal{M}_i^{D_p}$, then for each n, eventually the real copies of the $2n$th and $(2n+1)$st components of $\mathcal{G}$ will appear and remain forever in $\mathcal{M}_i^{D_p}$. Moreover, they will eventually be the oldest and lexicographically least copies in $\mathcal{M}_i^{D_p}$. Let l_n be the maximum true D_p-use on the edges in the loops in these $\mathcal{M}_i^{D_p}$ components. At this point, we will define $g_\alpha^{D_p}$ correctly on these components with a large use $\langle u_n, p\rangle$. Since $l_n < \langle u_n, p\rangle$, at most finitely many numbers enter D_p from the interval $(l_n, \langle u_n, p\rangle]$, but each time this happens, we define our map $g_\alpha^{D_p}$ to remain the same with the same use on the new oracle. Therefore, eventually our map $g_\alpha^{D_p}$ is never injured again on the $2n$th and $(2n+1)$st components. It follows that if $G \cong \mathcal{M}_i^{D_p}$, then $g_\alpha^{D_p}$ will be an embedding of $\mathcal{G}$ into $\mathcal{M}_i^{D_p}$ which will be an isomorphism by the structure of $\mathcal{G}$.

2.4. **Being not computably categorical relative to a degree.** Finally, for each $q \in P_1$ we want to make $\mathcal{G}$ not computably categorical relative to the c.e. set D_q. To achieve this, we build a D_q-computable copy $\mathcal{B}_q$ of $\mathcal{G}$ such that for all $e \in \omega$, the D_q-computable map $\Phi_e^{D_q} : \mathcal{G} \to \mathcal{B}_q$ is not an isomorphism. The graph $\mathcal{B}_q$ will be built globally.

Similarly to $\mathcal{G}$, we build the directed graph $\mathcal{B}_q$ in stages. At stage $s = 0$, we set $\mathcal{B}_q = \emptyset$. At stage $s > 0$, we add root nodes b_{2s}^q and b_{2s+1}^q to $\mathcal{B}_q$ and attach to each one a 2-loop. Next, we attach a $(5s+1)$-loop to b_{2s}^q and a $(5s+2)$-loop to b_{2s+1}^q with D_q-use s. However, throughout the construction, we may change the position of loops or add new loops to specific components of $\mathcal{B}_q$ depending on enumerations into A_q (and thus into D_q). For the $2s$th and $(2s+1)$st components of $\mathcal{B}_q$, we have three possible final configurations of the loops. If we never start the process of diagonalizing using these components, then they will remain the same forever:

$$b_{2s}^q : 2, 5s+1$$

$$b_{2s+1}^q : 2, 5s+2.$$

If we start, but don't finish, diagonalizing using these components, they will end in the following configuration:

$$b_{2s}^q : 2, 5s+1, 5s+3$$

$$b_{2s+1}^q : 2, 5s+1, 5s+4$$

If we complete a diagonalization with these components, then they will end as:

$$b^q_{2s} : 2, 5s + 1, 5s + 2, 5s + 4$$

$$b^q_{2s+1} : 2, 5s + 1, 5s + 2, 5s + 3.$$

For all $e \in \omega$, we meet the requirement

$$R^q_e : \Phi^{D_q}_e : \mathcal{G} \to \mathcal{B}_q \text{ is not an isomorphism.}$$

To satisfy this requirement, we will diagonalize against $\Phi^{D_q}_e$. Let α be an R^q_e-strategy.

When α is first eligible to act, it picks a large number n_α, and for the rest of this strategy, let $n = n_\alpha$. This parameter indicates which connected components of $\mathcal{B}_q$ will be used in the diagonalization. At future stages, α checks if $\Phi^{D_q}_e$ maps the $2n$th and $(2n + 1)$st connected component of $\mathcal{G}$ to the $2n$th and $(2n + 1)$st connected component of $\mathcal{B}_q$, respectively. If not, α does not take any action. If α sees such a computation, it defines m_α to be the max of the uses of the computations on each component and restrains $D_q \restriction m_\alpha + 1$.

At this point, our connected components in $\mathcal{G}[s]$ and $\mathcal{B}_q[s]$ are as follows:

$$a_{2n} : 2, 5n + 1 \qquad b^q_{2n} : 2, 5n + 1$$

$$a_{2n+1} : 2, 5n + 2 \qquad b^q_{2n+1} : 2, 5n + 2.$$

Since $\Phi^{D_q}_e$ looks like a potential isomorphism between $\mathcal{G}$ and $\mathcal{B}_q$, α will now take action to eventually force the true isomorphism to match a_{2n} with b^q_{2n+1} and to match a_{2n+1} with b^q_{2n} while preventing $\Phi^{D_q}_e$ from correcting itself on these components. Furthermore, α must do this in a way that will allow other requirements to succeed.

After m_α has been defined, α adds a $(5n + 3)$-loop to a_{2n} and a $(5n + 4)$-loop to a_{2n+1} in $\mathcal{G}[s]$. It also attaches a $(5n+3)$-loop to b^q_{2n} and a $(5n+4)$-loop to b^q_{2n+1} in $\mathcal{B}_q$[s]. Let v_α be a large unused number and set the use of all edges in these new loops appearing in $\mathcal{B}_q[s]$ to be $\langle v_\alpha, q\rangle$. Notice that $\langle v_\alpha, q\rangle > m_\alpha$ and that enumerating v_α into A_q will put $\langle v_\alpha, q\rangle$ into D_q, removing the $(5n + 3)$- and $(5n + 4)$-loops from $\mathcal{B}_q$ but not the $(5n + 1)$- or $(5n + 2)$-loops.

Our connected components in $\mathcal{G}[s]$ and in $\mathcal{B}_q[s]$ are now:

$$\begin{aligned} a_{2n} &: 2, 5n+1, 5n+3 \quad & b^q_{2n} &: 2, 5n+1, 5n+3 \\ a_{2n+1} &: 2, 5n+2, 5n+4 \quad & b^q_{2n+1} &: 2, 5n+2, 5n+4. \end{aligned}$$

After adding the $(5n+3)$- and $(5n+4)$-loops to both graphs, α must now wait for higher priority strategies which have already defined their maps on the $2m$th and $(2m+1)$st components for all $m \leq n$ to recover their maps before taking the last step. If β is a higher priority S or T^p strategy for which f_β or $g_\beta^{D_p}$ is already defined on the $2n$th and $(2n+1)$st components of $\mathcal{G}$, then before completing its diagonalization, α must wait for β to extend f_β or $g_\beta^{D_p}$ to be defined on the new $(5n+3)$- and $(5n+4)$-loops. We will refer to this action as α issuing a challenge to all higher priority S and T requirements. Moreover, α issues a challenge to all higher priority T requirements regardless of whether $p < q$, $q < p$, or if p and q are incomparable in P.

Once all higher priority strategies recover, α enumerates v_α into A_q (and thus $\langle v_\alpha, q\rangle$ goes into D_q). Doing this causes the $(5n+3)$-loop attached to b^q_{2n} and the $(5n+4)$-loop attached to b^q_{2n+1} to disappear in $\mathcal{B}_q[s]$. We now attach a $(5n+4)$-loop to b^q_{2n} and a $(5n+3)$-loop to b^q_{2n+1}. We also attach a $(5n+1)$-loop to a_{2n+1} and b^q_{2n+1} and a $(5n+2)$-loop to a_{2n} and b^q_{2n}, and we will refer to this process as homogenizing the components in $\mathcal{G}$ and in $\mathcal{B}_q$. The final configuration of our loops is:

$$\begin{aligned} a_{2n} &: 2, 5n+1, 5n+2, 5n+3 \quad & b^q_{2n} &: 2, 5n+1, 5n+2, 5n+4 \\ a_{2n+1} &: 2, 5n+1, 5n+2, 5n+4 \quad & b^q_{2n+1} &: 2, 5n+1, 5n+2, 5n+3. \end{aligned}$$

By homogenizing the components, we ensured that when we added loops for the diagonalization in $\mathcal{B}_q$, we also made adjustments in $\mathcal{G}$ to keep the components isomorphic to each other. Additionally, because $v_\alpha > m_\alpha$, the values $\Phi_e^{D_q}(a_{2n}) = b_{2n}$ and $\Phi_e^{D_q}(a_{2n+1}) = b_{2n+1}$ remain. So if $\Phi_e^{D_q}[s]$ is extended to a map on the entirety of $\mathcal{G}$, it cannot be a D_q-computable isomorphism, and so R^q_e is satisfied. If we meet R^q_e for all $e \in \omega$, we have that $\mathcal{G}$ is not computably categorical relative to D_q with $\mathcal{B}_q$ being the witness.

2.5. **Interactions between multiple strategies.** There are some interactions which can cause problems between these strategies. We will explain how the strategies described in this section, with some tweaks, can solve these issues.

We first point out that the independence requirements cause no serious issues for the other requirements. An N^p_e-strategy α, when it is first eligible to act, will pick a large unused number x_α, and so if it ever enumerates x_α into A_p, it will not violate any restraints placed by higher priority independence or R^q_e requirements. If this enumeration injures loops in or embeddings defined on components of $\mathcal{M}^{D_r}_i$ for some higher priority T^r_i-strategy where $p \leq r$, the T^r_i-strategy will be able to check for this D_r change when it is next eligible to act and will be able to react accordingly to succeed.

The next main interaction to note is between an R^q_e-strategy β and an S- or T-strategy α where $\alpha^\frown\langle\infty\rangle \subseteq \beta$. In the tree of strategies, $\alpha^\frown\langle\infty\rangle \subseteq \beta$ indicates that β guesses α will define an embedding of $\mathcal{G}$ into its graph $\mathcal{M}_i$ or $\mathcal{M}^{D_p}_i$. In the informal description for β, we had β wait for higher priority strategies to recover their embeddings defined on $\mathcal{G}$ after we added $(5n+3)$- and $(5n+4)$-loops to components of $\mathcal{G}$. When β adds the new loops, it updates α's parameter by setting $n_\alpha = n_\beta$ if α is an S-strategy. If α is instead a T-strategy, then β updates n_α to be the least $m \leq n_\beta$ such that $g^{D_p}_\alpha$ is no longer fully defined on the $2m$th and $(2m+1)$st components of $\mathcal{G}$ (for reasons that will become clear below). In either case, this causes α to return to previous components of $\mathcal{G}$ to find copies of them in either $\mathcal{M}_e$ or $\mathcal{M}^{D_p}_i$. If it is the case that either $\mathcal{G} \cong \mathcal{M}_e$ or $\mathcal{G} \cong \mathcal{M}^{D_p}_i$, α will eventually find copies and is able to define either a computable or D_p-computable isomorphism between the two graphs. Hence, the only tweaks needed for the S_e- and T^p_i-strategies α are steps in which they check if there is a lower priority R^q_e-strategy β where $\alpha^\frown\langle\infty\rangle \subseteq \beta$ that has issued its challenge after adding the new initial loops in $\mathcal{G}$.

There is a related technical point concerning R^q_e homogenizing the $2n_\beta$th and $(2n_\beta+1)$st components in the last step of its diagonalization. We do not ask the higher priority S-

and T-strategies α with $\alpha^\frown\langle\infty\rangle \subseteq \beta$ to go back and match these final homogenizing loops. Instead, we will extend f_α (or $g_\alpha^{D_p}$) to those homogenizing loops in a computable (or D_p-computable) way for α on the true path in the verification.

One last interaction arises between an R_e^q-strategy β and a T_i^p-strategy α where $\alpha^\frown\langle\infty\rangle \subseteq \beta$ and $q < p$ in P. With the current construction, we could have the following situation which makes it impossible for α to succeed. Suppose α finds copies at a stage s_0 of the $2n_\beta$th and $(2n_\beta+1)$st components of $\mathcal{G}[s_0]$ into $\mathcal{M}_i^{D_p}[s_0]$, and we have the following components in both graphs

$$a_{2n_\beta} : 2, 5n_\beta + 1 \qquad c : 2, 5n_\beta + 1$$

$$a_{2n_\beta+1} : 2, 5n_\beta + 2 \qquad d : 2, 5n_\beta + 2$$

where c and d are the root nodes of the copies of the $\mathcal{G}$ components found in $\mathcal{M}_i^{D_p}[s_0]$. At this point, α defines $g_\alpha^{D_p}[s_0]$ by mapping $a_{2n_\beta} \mapsto c$ and $a_{2n_\beta+1} \mapsto d$ with a use $\langle u_{\alpha,n_\beta}, p\rangle$.

Suppose at a later stage $s_1 > s_0$, β adds new loops to the corresponding components in $\mathcal{G}$ with a D_q-use of $\langle v_\alpha, q\rangle$ for v_α large and issues its challenge, and so our components in $\mathcal{G}[s_1]$ and $\mathcal{M}_i^{D_p}[s_1]$ are

$$a_{2n_\beta} : 2, 5n_\beta + 1, 5n_\beta + 3 \qquad c : 2, 5n_\beta + 1$$

$$a_{2n_\beta+1} : 2, 5n_\beta + 2, 5n_\beta + 4 \qquad d : 2, 5n_\beta + 2.$$

Then, suppose at stage $s_2 > s_1$ that $\Phi_i^{D_p}[s_1]$ adds the new loops correctly to c and d:

$$a_{2n_\beta} : 2, 5n_\beta + 1, 5n_\beta + 3 \qquad c : 2, 5n_\beta + 1, 5n_\beta + 3$$

$$a_{2n_\beta+1} : 2, 5n_\beta + 2, 5n_\beta + 4 \qquad d : 2, 5n_\beta + 2, 5n_\beta + 4.$$

Let z_{α,n_β} be the minimum use for any of the new edges in these loops, and assume that $\langle v_\alpha, q\rangle < z_{\alpha,n_\beta}$. The strategy α extends $g_\alpha^{D_p}[s_1]$ to map the $(5n_\beta+3)$- and $(5n_\beta+4)$-loops from $\mathcal{G}$ into $\mathcal{M}_i^{D_p}$ with a large use (i.e., greater than $\langle u_{\alpha,n_\beta}, p\rangle$). α has now met its challenge and takes the ∞ outcome.

Finally, suppose at a stage $s_3 > s_2$, β is eligible to act again. β enumerates v_α into A_q, which enumerates $\langle v_\alpha, q\rangle$ into D_q and D_p since $q < p$. Since $\langle v_\alpha, q\rangle \in D_q$, the $(5n_\beta+3)$- and $(5n_\beta+4)$-loops in $\mathcal{B}_q$ disappear, and so β can homogenize the $2n_\beta$th and $(2n_\beta+1)$st components in $\mathcal{G}$ and in $\mathcal{B}_q$ and diagonalize.

$$\begin{aligned} a_{2n_\beta} &: 2, 5n_\beta+1, 5n_\beta+2, 5n_\beta+3 \quad & b^q_{2n_\beta} &: 2, 5n_\beta+1, 5n_\beta+2, 5n_\beta+4 \\ a_{2n_\beta+1} &: 2, 5n_\beta+1, 5n_\beta+2, 5n_\beta+4 \quad & b^q_{2n_\beta+1} &: 2, 5n_\beta+1, 5n_\beta+2, 5n_\beta+3. \end{aligned}$$

The map $\Phi_e^{D_q}[s_3]$ still maps a_{2n_β} to $b^q_{2n_\beta}$ and $a_{2n_\beta+1}$ to $b^q_{2n_\beta+1}$ since the D_q-use for these computations is less than v_α and thus less than $\langle v_\alpha, q\rangle$.

However, because $\langle v_\alpha, q\rangle$ has gone into D_p as well and $\langle v_\alpha, q\rangle < z_{\alpha,n_\beta}$, the $(5n_\beta+3)$- and $(5n_\beta+4)$-loops in $\mathcal{M}_i^{D_p}[s_3]$ also disappear. But v_α was chosen to be large at stage s_1, so $v_\alpha > u_{\alpha,n_\beta}$ and so $g_\alpha^{D_p}[s_3]$ still maps a_{2n_β} to c and $a_{2n_\beta+1}$ to d. This allows the opponent controlling $\mathcal{M}_i^{D_q}$ to add loops in the following way to diagonalize:

$$\begin{aligned} a_{2n_\beta} &: 2, 5n_\beta+1, 5n_\beta+2, 5n_\beta+3 \quad & c &: 2, 5n_\beta+1, 5n_\beta+2, 5n_\beta+4 \\ a_{2n_\beta+1} &: 2, 5n_\beta+1, 5n_\beta+2, 5n_\beta+4 \quad & d &: 2, 5n_\beta+1, 5n_\beta+2, 5n_\beta+3. \end{aligned}$$

This now makes it impossible for α to succeed if $\mathcal{G} \cong \mathcal{M}_i^{D_p}$.

To solve this conflict, α needs to lift the use of $g_\alpha^{D_p}[s_1]$ when β starts its diagonalization process. Specifically, when β adds the $(5n_\beta+3)$- and $(5n_\beta+4)$-loops to $\mathcal{B}_q$ and sets their D_q-use to be $\langle v_\alpha, q\rangle$, we enumerate u_{α,n_β} into A_p, which puts $\langle u_{\alpha,n_\beta}, p\rangle$ into D_p but nothing into D_q. This action makes $g_\alpha^{D_p}$ undefined on the $2n_\beta$th and $(2n_\beta+1)$st components of $\mathcal{G}$. When α is next eligible to act, it will redefine $g_\alpha^{D_p}$ on the $2n_\beta$th and $(2n_\beta+1)$st components of $\mathcal{G}$ with a large use greater than $\langle v_\alpha, q\rangle$. Therefore, if β later enumerates v_α into A_q to diagonalize, the map $g_\alpha^{D_p}$ will become undefined on the entirety of the $2n_\beta$th and $(2n_\beta+1)$st components, preventing the opponent from using $\mathcal{M}_i^{D_p}$ to diagonalize against α.

It is possible that there is more than one T-strategy α with $\alpha^\frown\langle\infty\rangle \subseteq \beta$ associated with elements in P greater than q. In this case, β has to enumerate the use u_{α,n_β} for each such α

into A_q. These elements may cause $g_\alpha^{D_p}$ to become undefined on the $2m$th and $(2m+1)$st components for $m < n_\beta$, or for these components in $\mathcal{M}_\alpha^{D_p}$ to disappear. Therefore, for a T_i^p-strategy α with $\alpha^\frown\langle\infty\rangle \subseteq \beta$ and $q < p$, β resets n_α to be the least $m \leq n_\beta$ such that $g_\alpha^{D_p}$ no longer matches the $2m$th and $(2m+1)$st components in $\mathcal{G}$ and $\mathcal{M}_i^{D_p}$.

3. Proof of Theorem 1.5

In this section, we prove Theorem 1.5. Fix a computable partially ordered set $P = (P, \leq)$, and let $P = P_0 \sqcup P_1$ be a computable partition of P.

We build our computable directed graph $\mathcal{G}$ stage by stage as outlined in section 2.2, and for each $q \in P_1$, we build an isomorphic copy $\mathcal{B}_q$ of $\mathcal{G}$ as outlined in section 2.4.

We will also build a uniformly c.e. family of independent sets A_p for $p \in P$ via a priority argument on a tree of strategies. We define

$$D_p = \bigoplus_{q \leq p} A_q$$

and

$$\widehat{D_p} = \bigoplus_{q \neq p} A_q.$$

Our embedding h of P into the c.e. degrees is the map $h(p) = D_p$ for all $p \in P$.

3.1. Requirements.

Requirements. Recall our four types of requirements for our construction:

$$N_e^p : \Phi_e^{\widehat{D_p}} \neq A_p$$

$$S_e : \text{if } \mathcal{G} \cong \mathcal{M}_e\text{, then there exists a computable isomorphism } f_e : \mathcal{G} \to \mathcal{M}_e$$

$$T_i^p : \text{if } \mathcal{G} \cong \mathcal{M}_i^{D_p}\text{, then there exists a } D_p\text{-computable isomorphism } g_i^{D_p} : \mathcal{G} \to \mathcal{M}_i^{D_p}$$

$$R_e^q : \Phi_e^{D_q} : \mathcal{G} \to \mathcal{B}_q \text{ is not an isomorphism}$$

3.2. Construction.

Construction. Let $\Lambda = \{\infty <_\Lambda \cdots <_\Lambda w_2 <_\Lambda s <_\Lambda w_1 <_\Lambda w_0\}$ be the set of outcomes, and let $T = \Lambda^{<\omega}$ be our tree of strategies. The construction will be performed in ω many stages s.

We define the **current true path** π_s, the longest strategy eligible to act at stage s, inductively. For every s, λ, the empty string, is eligible to act at stage s. Suppose the strategy α is eligible to act at stage s. If $|\alpha| < s$, then follow the action of α to choose a successor $\alpha^\frown\langle o\rangle$ on the current true path. If $|\alpha| = s$, then set $\pi_s = \alpha$. For all strategies β such that $\pi_s <_L \beta$, initialize β (i.e., set all parameters associated to β to be undefined). If $\beta <_L \pi_s$ and $|\beta| < s$, then β retains the same values for its parameters.

We will now give formal descriptions of each strategy and their outcomes in the construction.

3.3. N_e^p-strategies and outcomes.

We first cover the N_e^p-strategies used to make each c.e. set A_p independent. Let α be an N_e^p-strategy eligible to act at stage s.

Case 1: If α is acting for the first time at stage s or has been initialized since the last α-stage, define its parameter x_α to be large, and take outcome w_0.

Case 2: If x_α is already defined and α took outcome w_0 at the last α-stage, check if

$$\Phi_e^{\widehat{D_p}}(x_\alpha)[s]\downarrow= 0.$$

If not, take the w_0 outcome. If $\Phi_e^{\widehat{D_p}}(x_\alpha)[s]\downarrow= 0$, enumerate x_α into A_p and take the s outcome which will preserve $\widehat{D_p} \upharpoonright (\mathrm{use}(\Phi_e^{\widehat{D_p}}(x_\alpha)[s]) + 1)$.

Case 3: If α took the s outcome the last time it was eligible to act and has not been initialized, take the s outcome again.

3.4. S_e-strategies and outcomes.

We now detail our S_e-strategy to make $\mathcal{G}$ computably categorical. Let α be an S_e-strategy eligible to act at stage s.

Case 1: If α is acting for the first time or has been initialized since the last α-stage, define $n_\alpha = 0$ and $f_\alpha[s]$ to be the empty map. Take the w_0 outcome.

Case 2: If α has defined n_α and is currently challenged by an R_e^q-strategy β with $\alpha^\frown\langle\infty\rangle \subseteq \beta$, then α acts as follows. In the verification, we show that there can only be one strategy challenging α at a time. When β challenged α, if the value of n_α was greater than n_β, then β redefined n_α to be equal to n_β. Therefore, we currently have $n_\alpha \leq n_\beta$. Furthermore, if

$n_\alpha = n_\beta$, then f_α may already be defined on the $(5n_\alpha+1)$- and $(5n_\alpha+2)$-loops in the $2n_\alpha$th and $(2n_\alpha+1)$st components of $\mathcal{G}$.

α searches for copies of the $2n_\alpha$th and $(2n_\alpha+1)$st components of $\mathcal{G}$ in $\mathcal{M}_e[s]$. More formally, if f_α is not defined on any loops in these components, then α searches for full copies of both components in $\mathcal{M}_e[s]$. If $n_\alpha = n_\beta$ and f_α is already defined on the $(5n_\alpha+1)$- and $(5n_\alpha+2)$-loops in $\mathcal{G}$, then α searches for the new loops of lengths $5n_\alpha+3$ and $5n_\alpha+4$ in the matched components in $\mathcal{M}_e[s]$. If no copies are found, set $f_\alpha[s] = f_\alpha[s-1]$, leave n_α unchanged, and take the w_{n_α} outcome. If copies are found, extend $f_\alpha[s-1]$ to $f_\alpha[s]$ by matching the components, increment n_α by 1 and check if $n_\alpha > n_\beta$ for this new n_α. If yes, take the ∞ outcome and declare β's challenge to have been met. If not, take the w_{n_α} outcome and let β's challenge remain active.

Case 3: If α has defined its parameter n_α and α is not currently challenged, then, α continues to search for copies of the $2n_\alpha$th and $(2n_\alpha+1)$st components of $\mathcal{G}$ in $\mathcal{M}_e[s]$. If no copies are found, define $f_\alpha[s] = f_\alpha[s-1]$, leave n_α unchanged, and take the w_{n_α} outcome. Otherwise, extend $f_\alpha[s-1]$ to $f_\alpha[s]$ by mapping the components to their respective copies in $\mathcal{M}_e[s]$, increment n_α by 1, and take the ∞ outcome.

3.5. **T_i^p-strategies and outcomes.** For each $p \in P_0$, we have the following T_i^p-strategy. Let α be a T_i^p-strategy eligible to act at stage s.

Case 1: If α is acting for the first time or has been initialized since the last α-stage, set $n_\alpha = 0$, define $g_\alpha^{D_p}[s]$ to be the empty function, and take the w_0 outcome.

Case 2: α is currently challenged by an R_e^q-strategy β where $\alpha^\frown\langle\infty\rangle \subseteq \beta$. Let s_0 be the stage at which β challenged α. In the verification, we will show that there can only be one strategy challenging α at a time. When β challenged α at stage s_0, it redefined n_α to be equal to the least $m \leq n_\beta$ such that $g_\alpha^{D_p}$ is not fully defined on the $2m$th and $(2m+1)$st components of $\mathcal{G}$ (for example, we may have that $m < n_\beta$ if an enumeration by α's challenge injures loops in $\mathcal{M}_i^{D_p}$).

If $q < p$ and s is the first α-stage since s_0 and n_α was greater than n_β at stage s_0, then we have to perform a preliminary action. In this case, β enumerated u_{α,n_β} into A_p, causing the map $g_\alpha^{D_p}[s_0]$ to become undefined on the $2n_\beta$th and $(2n_\beta+1)$st components of $\mathcal{G}$. Choose a new large number u_{α,n_β} and redefine $g_\alpha^{D_p}[s]$ to be equal to $g_\alpha^{D_p}[s_0]$ on the 2-loops, $(5n_\beta+1)$-loops, and $(5n_\beta+2)$-loops in these components with use $\langle u_{\alpha,n_\beta}, p\rangle$. These loops are still defined in $\mathcal{M}_i^{D_p}[s]$ since their D_p-uses remained the same and thus were below the old u_{α,n_β}. This ends the preliminary step.

Next, we perform the main action in this case. If $n_\alpha = n_\beta$ and α is already defined on the $(5n_\alpha+1)$- and $(5n_\alpha+2)$-loops of the $2n_\alpha$th and $(2n_\alpha+1)$st components in $\mathcal{G}$, then α searches for the oldest and lexicographically least copies of the $(5n_\alpha+3)$- and $(5n_\alpha+4)$-loops in $\mathcal{M}_i^{D_p}[s]$. If $g_\alpha^{D_p}$ is not currently defined on any of the loops in the $2n_\alpha$th and $(2n_\alpha+1)$st components of $\mathcal{G}$, then α searches for the oldest and lexicographically least copies of these components in $\mathcal{M}_i^{D_p}[s]$. In either case, if such copies are found, extend $g_\alpha^{D_p}[s]$ to map onto these copies with use $\langle u_{\alpha,n_\alpha}, p\rangle$ for large u_{α,n_α}, increment n_α by 1, and check if $n_\alpha > n_\beta$ for this new n_α. If yes, take the ∞ outcome and declare β's challenge to α to be met. If not, then take the w_{n_α} outcome and let β's challenge to α remain active.

Case 3: α is not currently challenged by an R_e^q-strategy. Let t be the last α-stage. In this case, α defined $g_\alpha^{D_p}[t]$ on the $2m$th and $(2m+1)$st components with use $\langle u_{\alpha,m}, p\rangle$ for $m < n_\alpha$. Let l_m be the max D_p-use for the computation of a loop in the image of the $2m$th and $(2m+1)$st components under $g_\alpha^{D_p}[t]$. In the verification, we will show that $l_m < u_{\alpha,m}$ for all $m < n_\alpha$.

Step 1: If there is an $m < n_\alpha$ such that $D_p[t] \upharpoonright l_m \neq D_p[s] \upharpoonright l_m$, then let m be the least such value. Note that for $m \leq m^* < n_\alpha$, the map $g_\alpha^{D_p}$ is now undefined on the $2m^*$th and $(2m^*+1)$st components of $\mathcal{G}$. The loops in the image of the $2k$th and $(2k+1)$st components of $\mathcal{G}$ under $g_\alpha^{D_p}[t]$ for $k < m$ remain in $\mathcal{M}_i^{D_p}[s]$. Update $n_\alpha = m$.

Step 2: By the update in Step 1, we have that for each $m < n_\alpha$ that $D_p[t] \upharpoonright l_m = D_p[s] \upharpoonright l_m$. For each $m < n_\alpha$, if any, where $D_p[t] \upharpoonright \langle u_{\alpha,m}, p\rangle \neq D_p[s] \upharpoonright \langle u_{\alpha,m}, p\rangle$, set $g_\alpha^{D_p}[s] = g_\alpha^{D_p}[t]$ on the loops in $\mathcal{G}$ in the $2m$th and $(2m+1)$st components with the same use as at stage t.

Step 3: We can now perform the main action of this case. α searches for the oldest and lexicographically least copies of the $2n_\alpha$th and $(2n_\alpha+1)$st components of $\mathcal{G}$ in $\mathcal{M}_i^{D_p}[s]$. If no copies are found, leave $g_\alpha^{D_p}$ and n_α unchanged and take outcome w_{n_α}. Otherwise, extend $g_\alpha^{D_p}$ by mapping the loops in the $2n_\alpha$th and $(2n_\alpha+1)$st components of $\mathcal{G}$ to their copies in $\mathcal{M}_i^{D_p}$ with use $\langle u_{\alpha,n_\alpha}, p\rangle$ where u_{α,n_α} is chosen large, increment n_α by 1, and take the ∞ outcome.

3.6. **R_e^q-strategies and outcomes.** Finally, for each $q \in P_1$, we have the following R_e^q-strategy. Let α be an R_e^q-strategy eligible to act at stage s. Recall that $\mathcal{B}_q$ is a D_q-computable copy of $\mathcal{G}$ which follows $\mathcal{G}$ on components not chosen by any R-strategy. For components chosen by α in particular, we have the following.

Case 1: If α is first eligible to act at stage s or has been initialized, define the parameter $n_\alpha = n$ to be large and take outcome w_0.

Case 2: If we are not in Case 1 and α took outcome w_0 at the last α-stage, check whether $\Phi_e^{D_q}[s]$ maps the $2n$th and $(2n+1)$st components of $\mathcal{G}$ isomorphically into $\mathcal{B}_q$. If not, take outcome w_0.

If so, set m_α to be the maximum D_q-use of these computations. Let v_α be large. Add a $(5n+3)$-loop to a_{2n} in $\mathcal{G}$ (computably) and to b_{2n}^q in $\mathcal{B}_q$ (with D_q-use $\langle v_\alpha, q\rangle$) and add a $(5n+4)$-loop to a_{2n+1} in $\mathcal{G}$ (computably) and to b_{2n+1}^q in $\mathcal{B}_q$ (with D_q-use $\langle v_\alpha, q\rangle$).

For each T_i^p-strategy γ where $\gamma^\frown\langle\infty\rangle \subseteq \alpha$ and $q < p$, enumerate the use $u_{\gamma,n}$ into A_p (and so $\langle u_{\gamma,n}, p\rangle$ enters D_p), and challenge γ. Note that if $n_\gamma < n_\alpha$, then there is no $u_{\gamma,n}$ to enumerate into A_p. For each S-strategy β where $\beta^\frown\langle\infty\rangle \subseteq \alpha$, challenge β and reset $n_\beta = n_\alpha$ if $n_\alpha > n_\beta$. Otherwise, leave n_α as it is. For each T-strategy β where $\beta^\frown\langle\infty\rangle \subseteq \alpha$, reset n_β to be the least $m \leq n_\alpha$ such that $g_\alpha^{D_p}$ does not match all of the $2m$th and $(2m+1)$st components of $\mathcal{G}$. Take outcome w_1.

Case 3: If α took outcome w_1 at the last α-stage, then enumerate v_α into A_q, move the $(5n+3)$-loop in $\mathcal{B}_q$ from b_{2n}^q to b_{2n+1}^q, and move the $(5n+4)$-loop in from b_{2n+1}^q to b_{2n}^q. Attach a $(5n+1)$-loop to a_{2n+1} and b_{2n+1}^q and a $(5n+2)$-loop to a_{2n} and b_{2n}^q. Let the D_q-use of these new loops in $\mathcal{B}_q$ be equal to the stage number s. Take outcome s.

Case 4: If α took outcome s at the last α-stage and has not been initialized, then take outcome s.

3.7. **Verification.** We first prove that the map h where $h(p) = D_p$ is an embedding into the c.e. degrees (if all N_e^p requirements are satisfied) and the computable and D_p-computable embeddings for $p \in P_0$, if they are defined, are the isomorphisms needed for $\mathcal{G}$'s categoricity. We will then prove key observations about the construction before stating the main verification lemma.

Lemma 3.1. *Suppose that for all $p \in P$ and $e \in \omega$, the requirement N_e^p is satisfied. Then, for all $p, q \in P$, we have that $p \leq q$ if and only if $D_p \leq_T D_q$.*

Proof. Assume that each N_e^p requirement was satisfied and suppose that $p \leq q$. Since $p \leq q$, for all r where $r \leq p$, we have that $r \leq q$ as well and so $D_p \leq_T D_q$. If $p \not\leq q$, then since

$$A_p \not\leq_T \bigoplus_{t \neq p} A_t,$$

it immediately follows that $A_p \not\leq_T \bigoplus_{t \leq q} A_t$ and hence $D_p \not\leq_T D_q$. □

Lemma 3.2. *If $f : \mathcal{G} \to \mathcal{G}$ is an embedding of $\mathcal{G}$ into itself, then f is an isomorphism (and is, in fact, the identity map).*

Proof. Let $f : \mathcal{G} \to \mathcal{G}$ be an embedding. Since embeddings preserve loops and only the root nodes a_m are contained in more than one loop, f must map root nodes to root nodes. Furthermore, since only the root nodes a_{2n} and a_{2n+1} can have $(5n+1)$-loops, we can only have that $f(a_{2n}) = a_{2n}$ or $f(a_{2n}) = a_{2n+1}$. However, the only situation in which a_{2n+1} has a $(5n+1)$-loop is when we attached a $(5n+3)$-loop to a_{2n} but *not* to a_{2n+1}. Thus, $f(a_{2n}) = a_{2n}$ and similarly, $f(a_{2n+1}) = a_{2n+1}$. Since $\mathcal{G}$ is a directed graph, it must map the loops attached to a_{2n} and to a_{2n+1} identically onto themselves. □

Lemma 3.3. *If $\mathcal{M}_e \cong \mathcal{G}$ for a computable directed graph $\mathcal{M}_e$ and $f_e : \mathcal{G} \to \mathcal{M}_e$ is an embedding defined on all of $\mathcal{G}$, then f_e is an isomorphism.*

Proof. This follows immediately from Lemma 3.2. □

By Lemma 3.3, if $\mathcal{M}_e \cong \mathcal{G}$ or $\mathcal{G} \cong \mathcal{M}_i^{D_p}$, then there is a unique isomorphism from $\mathcal{G}$ to $\mathcal{M}_e$ and from $\mathcal{G}$ to $\mathcal{M}_i^{D_p}$. We refer to the image of the $2n$th and $(2n+1)$st components of $\mathcal{G}$

in $\mathcal{M}_e^{D_p}$ as the **true copies** of these components in $\mathcal{M}_i^{D_p}$. We now prove several auxiliary lemmas about the construction.

Lemma 3.4. *If $\mathcal{G} \cong \mathcal{M}_i^{D_p}$, then for each n, there is an s such that for all $t \geq s$, the true copies of the $2n$th and $(2n+1)$st components of $\mathcal{G}$ in $\mathcal{M}_i^{D_p}$ are the oldest and lexicographically least isomorphic copies in $\mathcal{M}_i^{D_p}[t]$ of these components.*

Proof. Let u be the maximum D_p-use for the edges in the true copies of these components in $\mathcal{M}_i^{D_p}$ and let s_0 be such that $D_p \upharpoonright (u+1)[s_0] = D_p \upharpoonright (u+1)$. Because D_p is c.e., the true components will be defined at every stage $s \geq s_0$. There may be a finite number of older fake copies of these components, but they will disappear as numbers enter D_p, and so for a large enough $s \geq s_0$, the true copies will be the oldest and lexicographically least in $\mathcal{M}_i^{D_p}[s]$. □

Lemma 3.5. *Let α be an N_e^p-strategy that enumerates x_α into A_p at stage s. Unless α is initialized, no number below* $\mathrm{use}(\Phi_e^{\widehat{D_p}}(x_\alpha)[s])$ *is enumerated into $\widehat{D_p}$ after stage s.*

Proof. After α enumerates x_α into A_p at stage s, all strategies extending $\alpha^\frown\langle s\rangle$ will define new large parameters greater than $\mathrm{use}(\Phi_e^{\widehat{D_p}}(x_\alpha)[s])$. In particular, if there is an R-strategy $\beta \supseteq \alpha^\frown\langle s\rangle$, it will define its parameter n_β to be large. In the event that it issues a challenge to all higher priority S and T strategies $\gamma^\frown\langle\infty\rangle \subseteq \alpha^\frown\langle s\rangle \subseteq \beta$ on the $2n_\beta$th and $(2n_\beta+1)$st components, the use u_{γ,n_β}, if it exists, was chosen to be large (and so $u_{\gamma,n_\beta} > n_\beta$) on the mentioned components. Thus, $u_{\gamma,n_\beta} > \mathrm{use}(\Phi_e^{\widehat{D_p}}(x_\alpha)[s])$ also.

The only strategies which have parameters smaller than $\mathrm{use}(\Phi_e^{\widehat{D_p}}(x_\alpha)[s])$ are to the left of α on the tree of strategies or are R- or N-strategies β such that $\beta \subset \alpha$. When β enumerates a number into their assigned c.e. set, then it will take outcome s and initialize α. □

Lemma 3.6. *An S_e-strategy or a T_i^p-strategy can be challenged by at most one R-strategy at any given stage.*

Proof. Let α be an S_e-strategy (or T_i^p-strategy) and suppose that there exists some β such that $\alpha^\frown\langle\infty\rangle \subseteq \beta$ and β is an R-strategy that challenges α. If β challenges α at a stage s, then β takes the w_1 outcome for the first time. The strategies extending $\beta^\frown\langle w_1\rangle$ will choose witnesses at stage s, and in particular, none will challenge α. So, at most one strategy will challenge α at stage s. Until α is able to match the newly added loops in $\mathcal{G}$ to meet the challenge, α will take outcome w_{n_α}. R-strategies γ such that $\gamma \supseteq \alpha^\frown\langle w_{n_\alpha}\rangle$ will not challenge α since $w_{n_\alpha} \neq \infty$. □

Lemma 3.7. *Suppose α is an R^q_e-strategy that is never initialized after stage s. Then α can only challenge higher priority S-strategies and T-strategies at most once after stage s.*

Proof. Suppose α is an R^q_e-strategy that is never initialized after stage s and suppose it challenges all S-strategies and T-strategies β such that $\beta^\frown\langle\infty\rangle \subseteq \alpha$. Because α is never initialized again, if we ever return to α, it will take the s outcome as it can now diagonalize, and will continue to take the s outcome at all subsequent α-stages. If we do not return to α, α will not be able to challenge any higher priority S-strategies or T-strategies after stage s since it will never be eligible to act again. □

Lemma 3.8. *At most one strategy α enumerates numbers at any stage.*

Proof. Suppose numbers are enumerated at a stage s and α is the highest priority strategy which enumerates a number. α must either be for an R^q_e or an N^p_e requirement. If α is an N^p_e-strategy, it will take the s outcome for the first time, and if α is an R^q_e-strategy, it will either take the w_1 outcome or the s outcome for the first time. In either case, the remaining strategies which act at stage s will act by simply defining their parameters and taking the w_0 outcome. Therefore, α is the only strategy to enumerate a number at stage s. □

Lemma 3.9. *Let α be a T^p_i-strategy that defines $g^{D_p}_\alpha[s_m]$ on the $2m$th and $(2m+1)$st components of $\mathcal{G}$ at stage s_m. Until α is initialized (if ever), only strategies β such that $\alpha^\frown\langle\infty\rangle \subseteq \beta$ can enumerate a number $n \leq u_{\alpha,m}$ at any stage $t \geq s_m$.*

Proof. Let α be such a T^p_i-strategy. After defining $g^{D_p}_\alpha[s_m]$ on the $2m$th and $(2m+1)$st components of $\mathcal{G}$ at stage s_m, it takes the ∞ outcome. Hence, all strategies extending $\alpha^\frown\langle w_k\rangle$ for some k or to the right of α are initialized. These strategies will choose new parameters larger than $u_{\alpha,m}$ when they are next eligible to act. They are also the only strategies not extending $\alpha^\frown\langle\infty\rangle$ which can enumerate numbers without initializing α, so it suffices to show that they will not enumerate numbers below $u_{\alpha,m}$. Let β be such a strategy.

If β is an $N^{p'}_{e'}$-strategy, then it can only enumerate its parameter x_β into $A_{p'}$, and it was chosen such that $x_\beta > u_{\alpha,m}$. If β is an R^q_e-strategy then it enumerates two types of numbers: v_β and u_{γ,n_β} for any $T^{p'}_{i'}$-strategy such that $\gamma^\frown\langle\infty\rangle \subseteq \beta$ and $p' < q$. Since v_β will be chosen large after stage s_m, we have that $v_\beta > u_{\alpha,m}$. Since $\beta \supseteq \alpha^\frown\langle w_m\rangle$, we have that $n_\beta > u_{\alpha,m}$ because n_β was chosen to be large after $u_{\alpha,m}$ was defined. In particular, when n_β is chosen, γ cannot have defined u_{γ,n_β} since n_β is large, so when γ defines u_{γ,n_β} later, it must satisfy $u_{\gamma,n_\beta} > n_\beta$. Hence $u_{\gamma,n_\beta} > u_{\alpha,m}$. □

Lemma 3.10. *Let α be a T^p_i-strategy that takes a w_k outcome at a stage s. Let t be the next α-stage. Unless α has been initialized, $D_p[t] \upharpoonright \langle u_{\alpha,m}, p\rangle = D_p[s] \upharpoonright \langle u_{\alpha,m}, p\rangle$ for all $m < n_\alpha$, and so $g^{D_p}_\alpha[t]$ remains defined on the $2m$th and $(2m+1)$st components of $\mathcal{G}$ for all $m < n_\alpha$.*

Proof. This follows immediately from Lemma 3.9. □

Lemma 3.11. *Let α be a T^p_i-strategy. If α defines $g^{D_p}_\alpha$ on the $2m$th and $(2m+1)$st components of $\mathcal{G}$ at stage s, then $l_m[s] < u_{\alpha,m}[s]$ where $l_m[s]$ is the max use of the edges in the $(5m+1)$- and $(5m+2)$-loops in the copies of the $2m$th and $(2m+1)$st components of $\mathcal{G}$ in $\mathcal{M}^{D_p}_i$ and $\langle u_{\alpha,m}[s], p\rangle$ is the D_p-use of $g^{D_p}_\alpha[s]$ on these components. Furthermore, for all α-stages $t > s$, we have $u_{\alpha,m}[t] \geq u_{\alpha,m}[s] > l_m[s] = l_m[t]$ unless these components in $\mathcal{M}^{D_p}_i$ are injured or α is initialized.*

Proof. When α initially defines $g^{D_p}_\alpha$ on those components in **Case 2** or **Case 3** of its strategy, it chooses $u_{\alpha,m}[s]$ large, and so $u_{\alpha,m}[s] > l_m[s]$.

Consider an α-stage $t > s$ and assume α has not been initialized and the components in $\mathcal{M}^{D_p}_i$ remain intact. Since the D_p-use on the edges in the components remains the same, we have $l_m[t] = l_m[s]$, and so $(D_p \upharpoonright l_m[s])[t] = (D_p \upharpoonright l_m[s])[s]$. In particular, any update of the value of n_α in **Case 2** or in **Step 1** of **Case 3** of the T^p_i-strategy does not cause n_α to fall below m. Therefore, $u_{\alpha,m}[t]$ either has the same value as in the previous α-stage, or is updated by the preliminary action of **Case 2** (and so is chosen large), or is redefined in **Step 2** of **Case 3** (to its value at the previous α-stage). In all cases, $u_{\alpha,m}[t] \geq u_{\alpha,m}[s]$. □

Lemma 3.12. *Let α be an R^q_e-strategy that acts in **Case 2** at stage s by defining v_α. Let $t > s$ be the next α-stage and assume α is not initialized before t.*

1. *At stage t, for all S- and T-strategies β where $\beta^\frown\langle\infty\rangle \subseteq \alpha$, f_β and $g_\beta^{D_p}[t]$ is defined on the $2n_\alpha$th and $(2n_\alpha+1)$st components of $\mathcal{G}$. Furthermore, for T^p_i-strategies β, the minimum use of the computations $g_\beta^{D_p}[t]$ on the $(5n_\alpha+3)$- and $(5n_\alpha+4)$-loops is greater than v_α.*
2. *If $q < p$, then the D_p-use for $g_\beta^{D_p}[t]$ on a_{2n_α}, $a_{2n_\alpha+1}$, the $(5n_\alpha+1)$-loops, $(5n_\alpha+2)$-loops, and the 2-loops in $\mathcal{G}$ is greater than v_α.*

Proof. For (1), since α took the w_1 outcome at the last α-stage s and was not initialized after, it is now in **Case 3** at stage t. So in particular, it must be the case that α saw that for all S- and T-strategies β where $\beta^\frown\langle\infty\rangle \subseteq \alpha$ that their parameters n_β have exceeded n_α. In particular, if β is a T^p_i-strategy, it extended its map $g_\beta^{D_p}[s]$ on the $(5n_\alpha+3)$- and $(5n_\alpha+4)$-loops in $\mathcal{G}$ with a large use u_{β,n_α}. For each such β, we have that $u_{\beta,n_\alpha} > v_\alpha$. Furthermore, we claim that β cannot be challenged by another R-strategy before stage t. Suppose γ challenges β after β meets α's challenge. If $\beta^\frown\langle\infty\rangle \subseteq \gamma \subset \alpha$, then γ takes outcome w_1 when it challenges β, initializing α. If $\alpha \subseteq \gamma$, then γ cannot act until after stage t.

For (2), if β was a T^p_i-strategy where $q < p$ and $n_\beta > n_\alpha$, then α enumerated u_{β,n_α} into A_p, causing the map $g_\beta^{D_p}[s]$ to now be undefined on the entirety of the $2n_\alpha$th and $(2n_\alpha+1)$st components of $\mathcal{G}$. At stage t when α is eligible to act again, we have that β must have recovered its map on a_{2n_α}, $a_{2n_\alpha+1}$, the $(5n_\alpha+1)$-loops, $(5n_\alpha+2)$-loops, and the 2-loops in $\mathcal{G}$ with a new large use $u_{\beta,n_\alpha} > v_\alpha$.

On the other hand, if $n_\beta \leq n_\alpha$, when α challenged β, then $g_\beta^{D_p}$ was not yet defined on any of the loops in the $2n\alpha$th and $(2n_\alpha+1)$st components. Therefore, when β defines $g_\beta^{D_p}$ on these components, it uses a large use on every loop. □

Lemma 3.13. *Let α be an R^q_e-strategy that acts in **Case 2** at stage s by defining v_α and challenging higher priority S and T-strategies. If t is the next α-stage and α has not been initialized, then $D_q[t] \upharpoonright \langle v_\alpha, q\rangle + 1 = D_q[s] \upharpoonright \langle v_\alpha, q\rangle + 1$. In particular, all of the loops in the $2n_\alpha$th and $(2n_\alpha + 1)$st components of $\mathcal{B}_q$ remain intact.*

Proof. At stage s, α takes the w_1 outcome. By the proof of Lemma 3.8, no other strategy enumerates numbers at stage s. Since the strategies extending $\alpha^\frown\langle w_1\rangle$ and to the right of α choose new large parameters, none can enumerate a number below $\langle v_\alpha, q\rangle$ into D_q. If a strategy $\beta \subseteq \alpha$ enumerates a number, the path moves left, initializing α. Therefore, unless α is initialized, no number below $\langle v_\alpha, q\rangle$ can enter D_q before stage t. □

Lemma 3.14. *Let α be an R^q_e-strategy that acts in **Case 3** at stage s and takes the s outcome. Unless α is initialized, no number below the uses of the loops in the $2n_\alpha$th and $(2n_\alpha + 1)$st components of $\mathcal{B}_q$ or below m_q is enumerated into D_q after stage s.*

Proof. The proof of this lemma is almost identical to the proof of Lemma 3.5. □

We now state and prove the verification lemma for our construction.

Lemma 3.15 (Main Verification Lemma)**.** *Let $\pi = \liminf_s \pi_s$ be the true path of the construction, where π_s denotes the current true path at stage s of the construction. Let $\alpha \subset p$.*

1. *If α is an N^p_e-strategy, then there is an outcome o and an α-stage t_α such that for all α-stages $s \geq t_\alpha$, α takes outcome o where o ranges over $\{s, w_0\}$.*
2. *If α is an S_e-strategy, then either α takes outcome ∞ infinitely often or there is an outcome w_n and a stage $\hat{t}$ such that for all α-stages $s > \hat{t}$, α takes outcome w_n. If $\mathcal{G} \cong \mathcal{M}_e$, then α takes the ∞ outcome infinitely often and α defines a partial embedding $f_\alpha : \mathcal{G} \to \mathcal{M}_e$ which can be extended to a computable isomorphism $\hat{f}_\alpha : \mathcal{G} \to \mathcal{M}_e$.*
3. *Let α be a T^p_i-strategy. If $\mathcal{G} \cong \mathcal{M}^{D_p}_i$, then α takes the ∞ outcome infinitely often, and α defines a partial embedding $g^{D_p}_\alpha : \mathcal{G} \to \mathcal{M}^{D_p}_i$ which can be extended to a D_p-computable isomorphism $\hat{g}^{D_p}_\alpha : \mathcal{G} \to \mathcal{M}^{D_p}_i$.*
4. *If α is an R^q_e-strategy, then there is an outcome o and an α-stage t_α such that for all α-stages $s \geq t_\alpha$, α takes outcome o where o ranges over $\{s, w_1, w_0\}$.*

In addition, α satisfies its assigned requirement.

Proof. We first prove (1). Let $\alpha \subseteq \pi$ be an N_e^p-strategy and let s_0 be the least stage such that for all $s \geq s_0$, $\alpha \leq_L \pi_s$. Let x_α be its parameter at stage s_0. Suppose that at every α-stage $s \geq s_0$, either $\Phi_e^{\widehat{D_p}}(x_\alpha)[s] \uparrow$ or $\Phi_e^{\widehat{D_p}}(x_\alpha)[s] \downarrow \neq 0$. Then, α takes the w_0 outcome at every α-stage $s \geq s_0$. Furthermore, $\Phi_e^{\widehat{D_p}}(x_\alpha)$ either diverges or converges to a number other than 0. Since $x_\alpha \notin A_p$, α has met N_e^p.

Otherwise, there is some α-stage $t > s_0$ where $\Phi_e^{\widehat{D_p}}(x_\alpha)[t] \downarrow = 0$. At stage t, α enumerates x_α into A_p and takes the s outcome. Since α is never initialized, it takes the s outcome at every α-stage $s \geq t$. Furthermore, by Lemma 3.5 we have that

$$\Phi_e^{\widehat{D_p}}(x_\alpha) = \Phi_e^{\widehat{D_p}}(x_\alpha)[t] = 0 \neq 1 = A_p(x_\alpha),$$

and so N_e^p is satisfied.

For (2), let $\alpha \subseteq \pi$ be an S_e-strategy and let s_0 be the least stage such that for all $s \geq s_0$, $\alpha \leq_L \pi_s$. Suppose that α only takes the ∞ outcome finitely often. Fix an α-stage $s_1 > s_0$ such that α does not take the ∞ outcome at any α-stage $s \geq s_1$. Suppose that α takes the w_n outcome at stage s_1. There are two cases to consider.

If α is not challenged at stage s_1, then α acts as in **Case 3** of the S_e-strategy. Since α cannot be challenged by a requirement extending $\alpha^\frown\langle w_n\rangle$, α remains in **Case 3** at future α-stages unless it finds copies of the $2n$th and $(2n+1)$st components of $\mathcal{G}$ in $\mathcal{M}_e$. However, if it finds these copies, it would take the ∞ outcome, and so α must not ever find these copies. Hence, α takes the w_n outcome at every α-stage $s \geq s_1$. Moreover, $\mathcal{M}_e$ does not contain copies of the $2n$th and $(2n+1)$st components of $\mathcal{G}$ and so $\mathcal{M}_e$ is not isomorphic to $\mathcal{G}$.

If α is challenged at stage s_1, then α acts as in **Case 2** of the S_e-strategy. By a similar argument to the one above, α can never meet this challenge by finding images for the new loops in the $2n$th and $(2n+1)$st components of $\mathcal{G}$. Therefore, $\mathcal{M}_e \ncong \mathcal{G}$ and α takes the w_n outcome for all α-stages $s \geq s_1$.

From these two cases, it follows that if $\mathcal{G} \cong \mathcal{M}_e$, then α must take the ∞ outcome infinitely often. Let $n_\alpha[s]$ denote the value of n_α at the end of stage s (i.e., n_α could be possibly redefined by a challenging strategy during stage s). We claim that the value of $n_\alpha[s]$ goes to infinity as s goes to infinity. Suppose that α takes the ∞ outcome at a stage $s > s_0$. Either $n_\alpha[s] = n_\alpha[s-1]+1$ or $n_\alpha[s] = n_\beta[s]$ for some R-strategy β where $\alpha^\frown\langle\infty\rangle \subseteq \beta$ and $n_\beta[s] \leq n_\alpha[s-1]$. There can only be finitely many such β with $n_\beta[s] \leq n_\alpha[s-1]$. Once those strategies have challenged α (if ever), the value of n_α cannot drop below $n_\alpha[s-1]$ again. As α takes the ∞ outcome infinitely often, it follows that $n_\alpha[s]$ goes to infinity as s increases.

Let $f_\alpha = \bigcup\limits_{s\geq s_0} f_\alpha[s]$ be the limit of the partial $f_\alpha[s]$ embeddings for $s \geq s_0$. By Lemma 3.3, it remains to show that f_α can be computably extended to an embedding $\hat{f}_\alpha$ which is defined on all of $\mathcal{G}$. Note that no strategy $\beta \subseteq \alpha$ can add loops to $\mathcal{G}$ after stage s_0 or else the current true path would move to the left of α in the tree. Since $n_\alpha[s] \to \infty$ as $s \to \infty$, for each k, there is an α-stage s_k at which the n_α parameter starts with value k and α takes the ∞ outcome. At this stage, α found a copy of the $2k$th and $(2k+1)$st components of $\mathcal{G}$ (which consist of at least the initial set of loops) in $\mathcal{M}_e$ and defined $f_\alpha[s_k]$ on these loops. Therefore, f_α is defined on all of the initial loops attached to each a_{2k} and a_{2k+1}.

Only R-strategies β such that $\alpha^\frown\langle\infty\rangle \subseteq \beta$ can add loops to components on which f_α is already defined. If such a strategy β adds loops as in **Case 3** of its action, then it challenges α to find copies of these new loops. Since α takes the ∞ outcome infinitely often, it meets this challenge and extends f_α to be defined on the loops created by β. Then, β adds the homogenizing loops as in **Case 4** of its action. If $\mathcal{G} \cong \mathcal{M}_e$, then these homogenizing loops will have copies in $\mathcal{M}_e$. Suppose the $2n$th and $(2n+1)$st components of $\mathcal{G}$ have homogenizing loops, then because $\mathcal{G} \cong \mathcal{M}_e$, then only the nodes $f_\alpha(a_{2n})$ and $f_\alpha(a_{2n+1})$ in $\mathcal{M}_e$ will have copies of the respective homogenizing loops. So, we can computably extend f_α to $\hat{f}_\alpha$ by mapping the homogenizing loops to these copies, and thus $\hat{f}_\alpha$ is the computable isomorphism which satisfies the S_e requirement.

For (3), let $\alpha \subseteq \pi$ be a T_i^p-strategy and let s_0 be the least stage such that $\alpha \leq_L \pi_s$ for all $s \geq s_0$. If $\mathcal{G} \ncong \mathcal{M}_i^{D_p}$, then we satisfy the T_i^p requirement trivially. So, suppose $\mathcal{G} \cong \mathcal{M}_i^{D_p}$.

We claim that α takes the ∞ outcome infinitely often and that $n_\alpha[s] \to \infty$. By Lemma 3.4, the true copies of each pair of components will eventually appear in $\mathcal{M}_i^{D_p}$ and become the oldest copies of these components. If $g_\alpha^{D_p}$ maps the $2m$th and $(2m+1)$st components of $\mathcal{G}$ to fake copies in $\mathcal{M}_i^{D_p}$, then by Lemma 3.11, when a number less than $l_m[s]$ enters D_p to remove the fake copies, then the map $g_\alpha^{D_p}[s]$ also becomes undefined on those components. Therefore, for each n, α will eventually define $g_\alpha^{D_p}[s]$ correctly on the $2n$th and $(2n+1)$st components of $\mathcal{G}$, mapping them to the true copies in $\mathcal{M}_i^{D_p}$.

For the same reason, α will also meet each challenge after s_0 by an R-strategy extending $\alpha^\frown\langle\infty\rangle$. It follows that $n_\alpha[s] \to \infty$ as $s \to \infty$ and that $g_\alpha^{D_p} = \bigcup\limits_{s \geq s_0} g_\alpha^{D_p}[s]$ will correctly map all loops in $\mathcal{G}$ into $\mathcal{M}_i^{D_p}$ except the homogenizing loops added in **Case 3** of an R-strategy.

It remains to show that we can extend $g_\alpha^{D_p}$ in a D_p-computable way to an embedding $\hat{g}_\alpha^{D_p}$ defined on all of $\mathcal{G}$. Using the D_p oracle, we can tell when $g_\alpha^{D_p}[s]$ has correctly defined the original $(5m+1)$- and $(5m+2)$-loops from $\mathcal{G}$ into $\mathcal{M}_i^{D_p}$, as well as the $(5m+3)$- and $(5m+4)$-loops added (if ever) to $\mathcal{G}$ by an R-requirement. The D_p oracle will then tell us when the correct homogenizing loops show up in $\mathcal{M}_i^{D_p}$ (assuming that they were added to $\mathcal{G}$), so that we can extend $g_\alpha^{D_p}$ in a D_p-computable manner.

For (4), suppose $\alpha \subset \pi$ is an R_e^q-strategy and let $n_\alpha = n$ be its parameter. Let s_0 be the least stage such that $\alpha \leq_L \pi_s$ for all $s \geq s_0$. If α remains in the first part of **Case 2** from its description for all α-stages $s \geq s_0$, then R_e^q is trivially satisfied because $\Phi_e^{D_q}$ is not an isomorphism between $\mathcal{G}$ and $\mathcal{B}_q$, and α takes the w_0 outcome cofinitely often.

Otherwise, there is an α-stage $s_1 > s_0$ such that $\Phi_e^{D_q}[s_1]$ maps the $2n$th and $(2n+1)$st components of $\mathcal{G}$ isomorphically into $\mathcal{B}_q$. Then, α carries out all actions described in the second part of **Case 2**. In particular, it defines its target number v_α *after* enumerating all uses $u_{\gamma,n}$ (if they exist) for any T_i^p-strategy γ of higher priority with $q < p$ in P. α's challenge will eventually be met by all β such that $\beta^\frown\langle\infty\rangle \subset \alpha$ since $\beta^\frown\langle\infty\rangle \subset \pi$, and so let $s_2 > s_1$ be the next α-stage. By Lemma 3.13, $D_q[s_2] \upharpoonright \langle v_\alpha, q\rangle + 1 = D_q[s_1] \upharpoonright \langle v_\alpha, q\rangle + 1$, so the loops added to $\mathcal{B}_q$ at stage s_1 remain intact. At stage s_2, α enumerates v_α into A_q, moves the $(5n+3)$- and $(5n+4)$-loops in $\mathcal{B}_q$, and takes the s outcome at the end of this stage and at every future α-stage. By Lemma 3.12, for every T_i^p-strategy β with $\beta^\frown\langle\infty\rangle \subseteq \alpha$, we have that

- $g_\beta^{D_p}[s_2+1]$ is now undefined on the $(5n_\alpha+3)$- and $(5n_\alpha+4)$-loops in the $2n_\alpha$th and $(2n_\alpha+1)$st components of $\mathcal{G}$, and
- if $q < p$, then $g_\beta^{D_p}[s_2+1]$ is now undefined on the entirety of the $2n_\alpha$th and $(2n_\alpha+1)$st components of $\mathcal{G}$.

Since $\alpha \subset \pi$, α will never get initialized again. By Lemmas 3.13 and 3.14, we preserve $D_q \upharpoonright m_\alpha$. Recall that m_α is the use of the computation of $\Phi_e^{D_q}[s_1]$ where $\Phi_e^{D_q}[s_1](a_{2n}) = b_{2n}^q$ and $\Phi_e^{D_q}[s_1](a_{2n+1}) = b_{2n+1}^q$. So we have that $\Phi_e^{D_q} = \Phi_e^{D_q}[s_1]$, and so $\Phi_e^{D_q}(a_{2n}) = b_{2n}^q$ and $\Phi_e^{D_q}(a_{2n+1}) = b_{2n+1}^q$. However, a_{2n} is connected to a cycle of length $5n+3$ whereas b_{2n} is connected to a cycle of length $5n+4$, and so $\Phi_e^{D_q}$ cannot be a D_q-computable isomorphism between $\mathcal{G}$ and $\mathcal{B}_q$. □

4. Conclusion

The techniques used in the proof of Theorem 1.5 to make $\mathcal{G}$ computably categorical or not relative to a degree are compatible with techniques to create minimal pairs of c.e. degrees. In fact, we can show that we can embed the four element diamond lattice into the c.e. degrees in the following way.

Theorem 4.1. *There exists a computable computably categorical directed graph $\mathcal{G}$ and c.e. sets X_0 and X_1 such that*

(1) *X_0 and X_1 form a minimal pair,*
(2) *$\mathcal{G}$ is not computably categorical relative to X_0 and to X_1, and*
(3) *$\mathcal{G}$ is computably categorical relative to $X_0 \oplus X_1$.*

A natural question is whether we can embed bigger lattices into the c.e. degrees where elements of the lattice are labelled as either “c.c.” or “not c.c.” like in the poset case. We restrict to distributive lattices since those are embeddable into the c.e. degrees.

Question 4.2. *Let $L = (L, \wedge, \vee)$ be a computable distributive lattice and suppose we have a computable partition $L = L_0 \sqcup L_1$. Does there exist a computable computably categorical structure $\mathcal{A}$ and an embedding h of L into the c.e. degrees where $\mathcal{A}$ is computably categorical relative to each degree in $h(L_0)$ and is not computably categorical relative to each degree in $h(L_1)$?*

We end the paper by first discussing a restrictive case where our techniques do not work. Before we state this restrictive case, we observe that we can obtain a version of Theorem 1.5 where our graph $\mathcal{G}$ is not computably categorical using largely the same construction. Franklin and Solomon in [FS14] showed that every 2-generic degree $\mathbf{d}$ is low for isomorphism, that is, for every pair of computable structures $\mathcal{A}$ and $\mathcal{B}$, $\mathcal{A}$ is $\mathbf{d}$-computably isomorphic to $\mathcal{B}$ if and only if $\mathcal{A}$ is computably isomorphic to $\mathcal{B}$. So for every 2-generic $\mathbf{d}$, there exists *no* computable structure $\mathcal{A}$ where $\mathcal{A}$ is not computably categorical but is computably categorical relative to $\mathbf{d}$. This is also optimal in the generic degrees, meaning that for a 1-generic degree $\mathbf{c}$, we can apply techniques from this paper to build a computable structure that can change its behavior from being not computably categorical to being computably categorical relative to $\mathbf{c}$. A proof of this fact will appear in a sequel paper by Villano [Vil25].

It is still unknown how categoricity relative to a degree behaves above $\mathbf{0}'$ and strictly below $\mathbf{0}''$. A concrete question in this direction is the following.

Question 4.3. *For degrees in the the interval $[\mathbf{0}', \mathbf{0}'')$, is computable categoricity relative to a degree nonmonotonic like in the c.e. degrees?*

It is likely that categoricity relative to a degree also has nonmonotonic behavior in the interval $[\mathbf{0}', \mathbf{0}'')$, though a construction to witness this may require a $\mathbf{0}'''$-priority argument.

References

[Ash00] C. J. Ash. *Computable Structures and the Hyperarithmetical Hierarchy.* Ed. by J. Knight. New York: Elsevier, 2000.

[Ash+89] C. Ash et al. "Generic copies of countable structures". *Annals of Pure and Applied Logic* 42.3 (1989), 195–205. ISSN: 0168-0072.

[Chi90] J. Chisholm. "Effective Model Theory vs. Recursive Model Theory". *The Journal of Symbolic Logic* 55.3 (1990), 1168–1191. ISSN: 00224812.

[CHT17] B. F. Csima and M. Harrison-Trainor. "Degrees of Categoricity on a Cone via η-systems". *The Journal of Symbolic Logic* 82.1 (2017), 325–346. ISSN: 00224812, 19435886.

[DHTM21] R. Downey, M. Harrison-Trainor, and A. Melnikov. "Relativizing computable categoricity". *Proc. Amer. Math. Soc.* 149.9 (2021), 3999–4013. ISSN: 0002-9939.

[Erš77] J. L. Erš. "Theorie Der Numerierungen III". *Mathematical Logic Quarterly* 23.19-24 (1977), 289–371.

[FS14] J. N. Y. Franklin and R. Solomon. "Degrees that Are Low for Isomorphism". *Computability* 3 (2014), 73–89.

[GLS03] S. S. Gončarov, S. Lempp, and R. Solomon. "The computable dimension of ordered abelian groups". *Advances in Mathematics* 175.1 (2003), 102–143.

[Mar68] D. A. Martin. "The axiom of determinateness and reduction principles in the analytical hierarchy". *Bulletin of the American Mathematical Society* 74.4 (1968), 687 –689.

[Mon21] A. Montalbán. *Computable Structure Theory: Within the Arithmetic*. Perspectives in Logic. Cambridge University Press, 2021.

[Rem81] J. B. Remmel. "Recursively Categorical Linear Orderings". *Proc. Amer. Math. Soc.* 83.2 (1981), 387–391.

[Vil25] J. D. Villano. *Extensions of categoricity relative to a degree.* In preparation. 2025.

Department of Mathematics, University of Connecticut, Storrs, Connecticut 06269

Email address: javavill@uconn.edu