Classifications of computable structures

Karen Lange[*], Russell Miller[†], & Rebecca Steiner[‡]

June 28, 2014

Abstract

Let $\mathcal{K}$ be a family of structures, closed under isomorphism, in a fixed computable language. We consider effective lists of structures from $\mathcal{K}$ such that every structure in $\mathcal{K}$ is isomorphic to exactly one structure on the list. Such a list is called a *computable classification* of $\mathcal{K}$, up to isomorphism. Using the technique of Friedberg enumeration, we show that there is a computable classification of the family of computable algebraic fields, and that with a $\mathbf{0}'$-oracle, we can obtain similar classifications of the families of computable equivalence structures and of computable finite-branching trees. However, there is no computable classification of the latter, nor of the family of computable torsion-free abelian groups of rank 1, even though these families are both closely allied with computable algebraic fields.

MSC codes: 03C57, 03D45.

[*]Partially supported by grants numbered DMS-0802961 and DMS-1100604 from the National Science Foundation.

[†]Partially supported by grants numbered 64229-00 42 and 66582-00 44 from The City University of New York PSC-CUNY Research Award Program, and by a grant from the Queens College Research Enhancement Program.

[‡]The second and third authors were partially supported by grant numbered DMS-1001306 from the National Science Foundation. All authors wish to acknowledge Richard Shore for useful conversations and Matthew Harrison-Trainor, Alexander Melnikov, and Antonio Montalbán for allowing us to include Theorem 5.6 here.

1 Introduction

Classification of structures up to isomorphism is a common goal in all areas of mathematics. Here, following work of Goncharov and Knight in [8], we examine classification questions from the perspective of computable structure theory. Specifically, we are interested in *effective* classifications of fixed families of structures. Throughout, we examine families $\mathcal{K}$ of computable structures, closed under isomorphism, in a fixed language. (Recall that a structure $\mathcal{A}$ is *computable* if the atomic diagram of $\mathcal{A}$, denoted $\mathcal{D}(\mathcal{A})$, is computable. See [1] for general background on computable structure theory.) It is natural to ask which such families of structures have effective classifications.

Definition 1.1 Let $\mathcal{K}$ be a family of computable structures, closed under isomorphism, in a fixed computable language.

A *computable enumeration of* $\mathcal{K}$ consists of a computable function f such that, for every $n \in \omega$, $f(n)$ is a computable index of some structure in $\mathcal{K}$ (i.e. $\varphi_{f(n)} = \chi_{D(\mathcal{A})}$ for some $\mathcal{A} \in \mathcal{K}$), and for every $\mathcal{A} \in \mathcal{K}$, there is some $n \in \omega$ such that $f(n)$ is a computable index for a structure $\mathcal{M}$ isomorphic to $\mathcal{A}$ (i.e. $\varphi_{f(n)} = \chi_{D(\mathcal{M})}$ with $\mathcal{M} \cong \mathcal{A}$).

A *computable classification of* $\mathcal{K}$ is a computable enumeration of $\mathcal{K}$ such that each structure in $\mathcal{K}$ is isomorphic to exactly one structure in the enumeration.

(Our notation for partial computable functions φ_e and other computability concepts follows [16].) Many strongly minimal theories provide examples of families $\mathcal{K}$ with computable classifications.

Example 1.2 *Each of the following families has a computable classification.*

1. *Computable algebraically closed fields (either in a fixed characteristic or over all characteristics).*

2. *Computable vector spaces over a fixed computable field.*

3. *Computable successor structures, i.e. models of $Th(\mathbb{Z}, S)$ where S is the successor function on $\mathbb{Z}$.*

Goncharov and Knight [8] asked whether computable classifications exist for other families of structures, and in particular for the family of computable equivalence structures. (Among families not defined by strongly minimal theories, this family is widely considered to be one of the simplest possible examples.) Although this question remains open, they answered it for a subfamily of these structures, in [8, Theorem 5.5].

Theorem 1.3 (Goncharov & Knight [8]) *There is a computable classification of the family of computable equivalence structures with infinitely many infinite equivalence classes.*

Computable classification problems have natural connections to index set and isomorphism problems for families of computable structures. We refer the reader to [1] for background on index sets and isomorphism problems and to [2] for examples.

Definition 1.4 Let $\mathcal{K}$ be a family of computable structures, closed under isomorphism, in a fixed language.
(i) We define the *index set of* $\mathcal{K}$ to be the set $I(\mathcal{K})$ of computable indices for models in $\mathcal{K}$, i.e.

$$I(\mathcal{K}) = \{e \in \omega \mid (\exists \mathcal{A} \in \mathcal{K})\ [\varphi_e = \chi_{D(\mathcal{A})}]\}.$$

If $i \in I(\mathcal{K})$, let $\mathcal{A}_i$ be the (presentation of the) structure in $\mathcal{K}$ such that $\varphi_i = \chi_{D(\mathcal{A}_i)}$.
(ii) We call the set

$$\{\langle i, j\rangle \in \omega \mid i, j \in I(\mathcal{K})\ \&\ \mathcal{A}_i \cong \mathcal{A}_j\}$$

the *isomorphism problem of* $\mathcal{K}$.

In [8, Prop. 5.8], Goncharov and Knight observed the first statement below which places a restriction on the existence of computable classifications. We extend the first statement to a strong version of the second in Corollary 3.3.

Proposition 1.5

1. (Goncharov & Knight [8]) *No family $\mathcal{K}$ whose isomorphism problem is Σ^1_1-complete and whose index set is hyperarithmetic has a computable classification, nor even a hyperarithmetic classification.*

2. *For each $n \in \omega$, no family $\mathcal{K}$ whose whose index set is Δ^0_n and whose isomorphism problem is Σ^0_n-complete has a computable classification.*

Part 1 of Proposition 1.5 yields many examples of families lacking computable classifications.

Example 1.6 *The following families of computable structures do not have computable classifications.*

1. *Graphs*
2. *Fields*
3. *Any of the families described by Hirschfeldt, Khoussainov, Shore, and Slinko in [10], including computable partial orders, lattices, rings, groups, and integral domains.*

In Corollary 3.3, we use an extension of Part 2 of Proposition 1.5 to show that there is no computable classification of computable torsion-free abelian groups of rank 1. In contrast, for families $\mathcal{K}$ whose isomorphism problem is Π^0_n-complete (for some $n \in \omega$), the only way we have found to prove that $\mathcal{K}$ has no computable classification is to show that it does not even have a computable enumeration; this method is used in §4 and §5.1.

1.1 Our methods

Friedberg's 1958 article [6] provides our main technique for establishing positive results on classifications. There Friedberg answered three significant questions in classical computability. Two of his results, the existence of a maximal c.e. set and the Friedberg splitting construction, have had major impact on the discipline. The third, the existence of a computable enumeration of all c.e. sets without repetition, has come to be viewed more as a curiosity, at least within pure computability theory: the enumeration given by Friedberg is not an *acceptable* enumeration (in the terminology of [16, I.5.9]), and fails to satisfy such basic results as the s-m-n Theorem, making this enumeration less useful. The question addressed by Friedberg was natural, and other researchers have studied whether such enumerations exist for other families of sets of a given complexity (see for example [9]), but Friedberg's answer has had no fundamental impact on the study of computably enumerable degrees. In contrast, in computable structure theory, researchers have used Friedberg's technique to address questions related to those we consider. This study of *numberings* or *enumerations* has taken place mainly in the former Soviet Union; see for example [5].

From the perspective of computable structure theory, Friedberg's result can be thought of as finding a computable classification of the Σ^0_1-definable subsets of $\mathbb{N}$, up to equality. We will apply Friedberg's approach, using his techniques to prove the existence of certain effective classifications. In this paper, we focus on four families of computable structures: algebraic fields, torsion-free abelian groups of rank 1, finite-branching trees, and equivalence structures. Algebraic fields prove to be tractable using the Friedberg method, and this leads us to consider the related families of abelian groups and trees. We find Friedberg's technique useful for the trees as well, and for computable equivalence structures, but to apply it there, we need more computational power. Therefore, we relativize Definition 1.1 to other Turing degrees, in order to develop a fuller understanding of effective classifications for these families.

Definition 1.7 Let $\boldsymbol{d}$ be a Turing degree and C a subset of ω. We say $e \in \omega$ is a *C-computable index* for a structure $\mathcal{A}$ if $\Phi_e^C = \chi_{D(\mathcal{A})}$. A *$\boldsymbol{d}$-computable classification of $\mathcal{K}$ by C-computable indices* is a uniformly $\boldsymbol{d}$-computable enumeration of C-computable indices for structures in $\mathcal{K}$ such that each structure in $\mathcal{K}$ is represented exactly once in the enumeration up to isomorphism.

In all cases we will consider in this article, either $C = \emptyset$ or $C \in \boldsymbol{d}$. That is, we study $\boldsymbol{d}$-computable classifications either by computable indices or by $\boldsymbol{d}$-computable indices. For us, $\boldsymbol{d}$ will always be of the form $\mathbf{0}^{(n)}$. Notice that in the definition, when $C \equiv_T \widetilde{C}$, a number e may be a C-computable index for $\mathcal{A}$ without being a $\widetilde{C}$-computable index for $\mathcal{A}$. However, in this situation, there are computable total injective functions f and $\tilde{f}$ such that, for every such e and $\tilde{e}$, $\Phi_e^C = \Phi_{f(e)}^{\widetilde{C}}$ and $\Phi_{\tilde{e}}^{\widetilde{C}} = \Phi_{\tilde{f}(\tilde{e})}^{C}$. Hence it is reasonable to speak of $\boldsymbol{d}$-computable indices without specifying the oracle set in $\boldsymbol{d}$.

1.2 Families to be studied

We consider algebraic fields in §2, torsion-free abelian groups in §3, finite-branching trees in §4, and equivalence structures in §5.

1.2.1 Algebraic fields

Since the isomorphism problem for computable fields of characteristic 0 is Σ^1_1-complete and the index set of such fields is only Π^0_2, Proposition 1.5 gives the following result.

Proposition 1.8 *There is no hyperarithmetic classification of all computable fields.* ■

However, when we restrict ourselves to algebraic computable fields, we fare much better. Recall that a field is *algebraic* if every element of the field satisfies a nonzero polynomial over the prime subfield (which is either $\mathbb{Q}$ or $\mathbb{F}_p$, depending on the characteristic of the field). In §2.2, we will use Friedberg's method to prove the following theorem.

Theorem 1.9 *There is a computable classification of the family of computable algebraic fields.*

1.2.2 Torsion-free abelian groups of rank 1

We write $\mathbf{TFAb}_1$ for the family of computable torsion-free abelian groups of rank 1. These are precisely the c.e. subgroups of a computable presentation of the group $(\mathbb{Q}, +)$, which allows us to enumerate them computably. However, we show that there is no computable classification of $\mathbf{TFAb}_1$. Indeed, we prove that $\mathbf{0}^{(n)}$-computable classifications of $\mathbf{TFAb}_1$ by computable indices exist only for $n \geq 3$.

Theorem 1.10 *There is no $\mathbf{0}''$-computable classification of $\boldsymbol{TFAb}_1$ by computable indices, but there does exist a $\mathbf{0}'''$-computable classification of $\boldsymbol{TFAb}_1$ by computable indices.*

We prove the existence portion of Theorem 1.10 in Lemma 3.4 and the nonexistence portion in Corollary 3.3.

1.2.3 Finite-branching trees

For our purposes, a *tree* T is a substructure of the structure $\omega^{<\omega}$ of all finite strings of natural numbers. The language contains just a unary function, the *predecessor function* P, which names the immediate predecessor of each element (and maps the root to itself). Note that P is computable on $\omega^{<\omega}$. To be a tree, T must be nonempty and closed under P, and to be computable, T must be a computable subset of $\omega^{<\omega}$. (Our discussion does not necessarily carry over to computable trees in the language of partial orders.) Finally, T

is *finite-branching* if, for each $x \in T$, the pre-image of x under P is finite. We use $\mathcal{T}$ to denote the family of all computable finite-branching trees.

Finite-branching trees are algebraic, in the model-theoretic sense of the word, and have been shown in [17] to have properties very similar to those of algebraic fields. However, we will show that there is no computable enumeration of $\mathcal{T}$, let alone any computable classification of this family. (In fact, we show more; see Proposition 4.3). On the other hand, in Proposition 4.5 we will use Friedberg's method to give a classification of $\mathcal{T}$ using a $\mathbf{0}'$-oracle, in a way which is not known to be possible for $\mathbf{TFAb}_1$. The theorem below follows from Propositions 4.3, 4.4, and 4.5.

Theorem 1.11 *There exists a $\mathbf{0}'''$-computable classification of $\mathcal{T}$ by computable indices as well as a $\mathbf{0}'$-computable classification of $\mathcal{T}$ by $\mathbf{0}'$-computable indices. However, no classification of $\mathcal{T}$ by computable indices can be $\mathbf{0}''$-computable.*

1.2.4 Equivalence structures

Goncharov and Knight examined computable equivalence structures in [8], as noted above. (A countable *equivalence structure* is simply an equivalence relation on the domain ω.) They gave a computable classification of the family $\mathcal{E}_\infty$ of all computable equivalence structures that contain infinitely many infinite equivalence classes. The same problem for the family $\mathcal{E}_n$ of computable equivalence structures with exactly n infinite classes proves thornier, and we show in §5.1 that there is no computable enumeration (let alone classification) of any family $\mathcal{E}_n$. However, in §5.2, we produce a $\mathbf{0}'$-computable classification of $\mathcal{E}_0$ using $\emptyset'$-computable indices, applying Friedberg's method once again, relativized to a $\mathbf{0}'$-oracle and starting with a particular $\mathbf{0}'$-computable enumeration of $\mathcal{E}_0$. From this result, we readily produce a $\mathbf{0}'$-computable classification of the entire family $\mathcal{E}$ of all computable equivalence relations, again using $\emptyset'$-computable indices. The oracle $\mathbf{0}'$ is not particularly powerful – far stronger oracles are unable to produce classifications of many standard families with Σ^1_1-complete isomorphism problems, as shown in Proposition 1.5 – and so we regard this result as a vindication of the view that computable equivalence structures, while nontrivial, are not a particularly complex family of structures.

2 Fields by Friedberg

After discussing some necessary background in §2.1, we construct a computable classification of the family of algebraic fields in §2.2, proving Theorem 1.9.

2.1 Background on fields

Recall that the *splitting set* S_F of a computable field F is the set of reducible polynomials in $F[X]$. (Formally, it is the set of code numbers for these polynomials when $F[X]$ is listed out in the canonical way from the computable presentation of F.) The Turing degree of the splitting set does not vary between computable presentations of a single algebraic field, and S_F is Turing-equivalent to the *root set* R_F, the set of those polynomials in $F[X]$ having roots in F. If S_F is computable, then F is said to have a *splitting algorithm*, and this algorithm allows one to identify the irreducible polynomials in $F[X]$. Finally, there is a computable presentation of the algebraic closure $\overline{F}$ of F: this presentation may be given uniformly in an index for F, as may an index for a computable embedding $g : F \to \overline{F}$, and the image $g(F)$ of F within $\overline{F}$ is Turing-equivalent to S_F. Hence $g(F)$ is computable if and only if F has a splitting algorithm. All of this follows essentially from Rabin's Theorem (see [15]).

We take advantage of the following facts.

Lemma 2.1 *For each characteristic $p \geq 0$, there is a computable enumeration $\langle F_e \rangle_{e \in \omega}$ of all computable algebraic fields of characteristic p.*

Proof. Fix a computable presentation $\overline{Q}$ of the algebraic closure of the prime field Q ($= \mathbb{Q}$ or $= \mathbb{F}_p$) of characteristic p. For each e, let F_e be the subfield of $\overline{Q}$ generated by the c.e. set W_e. Thus, each F_e is itself c.e., uniformly in e, and the fields F_e form a computable enumeration of all computably presentable algebraic fields of characteristic p (since every such field has a computable embedding into $\overline{Q}$, with c.e. image). Notice that, while F_e itself may not be technically a computable field (if its domain, which is c.e., fails to be computable), it is computably isomorphic to a computable field, just by taking a 1-1 computable enumeration of its elements and pulling back the field operations to the domain of this enumeration. Of course, in positive characteristic, we allow finite computable fields in our enumeration. ∎

The following lemma appears as [14, Corollary 3.9], and also (with a different proof) in [7, Appendix A]. Essentially it follows from König's Lemma.

Lemma 2.2 *Two algebraic fields E and F of characteristic 0 are isomorphic if and only if, for all finitely generated algebraic field extensions K of $\mathbb{Q}$, the field K embeds in E if and only if K embeds in F.*

Now we are prepared to do the Friedberg construction with computable algebraic fields. Formally (and in more generality), this appears as Theorem 2.3 below. When this theorem is applied to fields, the fields F_e play the role of the c.e. sets R_e in Friedberg's article (or W_e, in modern notation). Isomorphism $F_i \cong F_j$ is a Π^0_2 property, by Lemma 2.2, just as is equality $W_i = W_j$ of c.e. sets. We think of F_i and F_j (or finitely generated subfields thereof) as *agreeing up to k* if the subfield $F_i \restriction k$ generated by the first k elements of F_i embeds into F_j and likewise $F_j \restriction k$ embeds into F_i. Thus, by Lemma 2.2, $F_i \cong F_j$ if and only if F_i and F_j agree up to every $k \in \omega$. Where Friedberg puts a new large number into some c.e. set in his enumeration (to distinguish it from another set), we can adjoin a d-th root of unity to one of our fields, for a new large prime d. The Friedberg construction then gives a computable classification of computable algebraic fields of a given characteristic. The next subsection gives the full proof.

2.2 Friedberg's Construction

We now recast Friedberg's construction of a classification of all c.e. sets in terms of classifying some family of $\boldsymbol{d}$-computably presentable structures of a given kind.

Given a structure $\mathcal{M}$ with domain $\subseteq \omega$, we let $\mathcal{M} \restriction s$ be the substructure of $\mathcal{M}$ generated by the elements $\{0, 1, \ldots, s-1\} \cap \text{dom}(\mathcal{M})$ under the function symbols in the language. Since we allow function symbols, $\mathcal{M} \restriction s$ need not be finite. In general its domain may only be computably enumerable, but we treat it as an $\mathcal{M}$-computable structure, since we get an $\mathcal{M}$-computable isomorphism from each $\mathcal{M} \restriction s$ onto a computable structure, uniformly in s, by mapping an initial segment of ω onto the domain of $\mathcal{M} \restriction s$. We do specifically allow $\mathcal{M}$ to have finite domain; this is important when dealing with fields in positive characteristic, and also for equivalence structures in §5.2. We also say $\mathcal{M}_i \restriction s$ is a *proper substructure* of $\mathcal{M}_j \restriction t$ if the former embeds into the latter but they are not isomorphic. (An *embedding* is just an injective homomorphism.)

Theorem 2.3 *Let $\boldsymbol{d}$ be a Turing degree, and $\mathcal{K}$ a family of structures, closed under isomorphism, in a fixed $\boldsymbol{d}$-computable language. Suppose there exists a $\boldsymbol{d}$-computable enumeration $\langle \mathcal{M}_i \rangle_{i \in \omega}$ of $\mathcal{K}$ by $\boldsymbol{d}$-computable indices satisfying the following conditions.*

1. *For each $\mathcal{M}_i$ and each stage s,*
   (a) *$\mathcal{M}_i \restriction s$ is an element of $\mathcal{K}$ and*
   (b) *there is some $t > s$ and $j \in \omega$ such that*
      - *$\mathcal{M}_i \restriction s$ is a proper substructure of $\mathcal{M}_j \restriction t$ and*
      - *for all $k < s$, $\mathcal{M}_j \restriction t$ is not isomorphic to $\mathcal{M}_k \restriction s$.*
2. (a) *For every two indices i and j, $\mathcal{M}_i \cong \mathcal{M}_j$ iff i and j satisfy:*

   $$(\forall s)(\exists t)\ [\mathcal{M}_i \restriction s \textit{ embeds into } \mathcal{M}_j \restriction t \ \ \& \ \ \mathcal{M}_j \restriction s \textit{ embeds into } \mathcal{M}_i \restriction t].$$

   (b) *The following two sets are both $\boldsymbol{d}$-computable.*

   $$\{\langle i,t,j,s\rangle : \mathcal{M}_i \restriction t \cong \mathcal{M}_j \restriction s\}$$
   $$\{\langle i,t,j,s\rangle : \mathcal{M}_i \restriction t \textit{ embeds into } \mathcal{M}_j \restriction s\}$$

   Thus, the isomorphism problem and the proper substructure problem for any two structures $\mathcal{M}_i \restriction t$ and $\mathcal{M}_j \restriction s$ are $\boldsymbol{d}$-computable.

Then there is a $\boldsymbol{d}$-computable classification by $\boldsymbol{d}$-computable indices of the structures in $\mathcal{K}$.

Proof. Let $\langle \mathcal{M}_i \rangle_{i \in \omega}$ be a $\boldsymbol{d}$-computable enumeration of all structures in a family $\mathcal{K}$ by $\boldsymbol{d}$-computable indices satisfying the assumptions listed in the theorem. We construct a $\boldsymbol{d}$-computable classification $\langle \mathcal{N}_i \rangle_{i \in \omega}$ by $\boldsymbol{d}$-computable indices of the structures in $\mathcal{K}$ by employing Friedberg's method. For the reader's convenience, we imitate Friedberg's original construction in [6, Thm. 3] as closely as possible. In particular, at times we will assign $\mathcal{N}_k$ to be a *follower* of some $\mathcal{M}_i$. At stages s when $\mathcal{N}_k$ is following $\mathcal{M}_i$ we construct $\mathcal{N}_{k,s}$ to be isomorphic to $\mathcal{M}_i \restriction s$. If at any point we *release* $\mathcal{N}_k$ as a *follower* of $\mathcal{M}_i$, we call $\mathcal{N}_k$ *free* and $\mathcal{N}_k$ will never again be assigned to follow any other $\mathcal{M}_j$. However, $\mathcal{M}_i$ can be assigned a new follower at a later stage. By Assumption (1a), for all $j \in \omega$, the structure $\mathcal{M}_j \restriction 0$ generated by the empty set lies in $\mathcal{K}$; we consider in Corollary 2.7 below how to amend this assumption. At each stage s, we take action for some $\mathcal{M}_i$. We let e_s denote the index i of the $\mathcal{M}_i$ for which we take action at stage s. Specifically, we set e_s equal to the number of prime factors of s. This definition ensures that we take action for each $\mathcal{M}_e$ at infinitely many stages during the construction.

Assumptions (2a) and (2b) imply that $\mathcal{M}_i$ and $\mathcal{M}_j$ being isomorphic is a $\Pi_2^{\boldsymbol{d}}$-property. In particular, we may define a $\boldsymbol{d}$-computable *chip function* $c(i,j,s)$ as follows:

$$c(i,j,s) = \begin{cases} 0 & \text{if } s = 0. \\ c(i,j,s-1)+1 & \mathcal{M}_i{\upharpoonright}t \text{ embeds into } \mathcal{M}_j{\upharpoonright}s \text{ and} \\ & \mathcal{M}_j{\upharpoonright}t \text{ embeds into } \mathcal{M}_i{\upharpoonright}s, \\ & \text{where } t = c(i,j,s-1),\ s > 0. \\ c(i,j,s) = c(i,j,s-1) & \text{otherwise.} \end{cases}$$

In other words, $c(i,j,s)$ "gives a chip" to the pair (i,j) at stage s (i.e. outputs $c(i,j,s) = c(i,j,s-1)+1$) if and only if the stage t approximations of $\mathcal{M}_i$ and $\mathcal{M}_j$ embed into each other's stage s approximations, where t is the total number of chips received by the pair (i,j) at all stages less than s. This definition is symmetric in i and j, and the pair (i,j) receives infinitely many chips (over all stages s) if and only if $\mathcal{M}_i \cong \mathcal{M}_j$.

We will see by induction that the construction satisfies the following assumption: that for each stage $t < s$ and each $\mathcal{N}_i$ (which may be a follower or free at stage t), we $\boldsymbol{d}$-computably know a $\boldsymbol{d}$-computable index e' and stage t' such that $\mathcal{N}_{i,t} \cong \mathcal{M}_{e'}{\upharpoonright}t'$. Hence, by Assumption (2b), it is $\boldsymbol{d}$-computable to determine whether a given $\mathcal{N}_{i,t}$ is isomorphic to a given $\mathcal{M}_{\hat{e}}{\upharpoonright}\hat{t}$. We may also inductively assume that the current follower $\mathcal{N}_k$ for $\mathcal{M}_{e_s}$ at the beginning of stage s, if any, satisfies $\mathcal{N}_{k,t} = \mathcal{N}_{k,s-1} \cong \mathcal{M}_{e_s}{\upharpoonright}t$ for some $t < s$. (In other words, $\mathcal{N}_k$ has not changed since stage t.)

At stage s, having fixed e_s, we have three cases.

Case 1 ($\mathcal{M}_{e_s}$ **with follower appears isomorphic to earlier** $\mathcal{M}_e$**.**) Suppose that $\mathcal{M}_{e_s}$ has a follower $\mathcal{N}_k$ and that there exists an $e < e_s$ with $c(e, e_s, s) \geq k$. Then we release $\mathcal{N}_k$ as a follower of $\mathcal{M}_{e_s}$.

Case 2 (**For some** k **with additional properties,** $\mathcal{N}_{k,s-1} \cong \mathcal{M}_{e_s}{\upharpoonright}s$**.**)

If Case 1 does not hold, and there exists a k such that $\mathcal{N}_{k,s-1} \cong \mathcal{M}_{e_s}{\upharpoonright}s$ with one of the following properties:

- $\mathcal{N}_k$ is the follower of $\mathcal{M}_e$ for some $e \leq e_s$; or

- $\mathcal{N}_k$ is not currently a follower of any $\mathcal{M}_e$, and either $k \leq e_s$ or $\mathcal{N}_k$ was previously *displaced by* $\mathcal{M}_{e_s}$ via Case 3,

then we do nothing.
Case 3 (**Case 1 and Case 2 do not hold.**). If Case 1 and Case 2 do not hold, we execute the following three steps.

1. (**Ensure $\mathcal{M}_{e_s}$ has a follower.**)
   If $\mathcal{M}_{e_s}$ has no follower, assign $\mathcal{N}_k$ to follow $\mathcal{M}_{e_s}$ where k is the least index for which $\mathcal{N}_k$ has never yet been a follower, and build $\mathcal{N}_{k,s}$ isomorphic to $\mathcal{M}_{e_s}\restriction s$.

2. (**Update any existing follower for $\mathcal{M}_{e_s}$.**)
   If $\mathcal{M}_{e_s}$ already had a follower $\mathcal{N}_k$ at stage $(s-1)$, then add elements to $\mathcal{N}_{k,s-1}$ as needed so that $\mathcal{N}_{k,s} \cong \mathcal{M}_{e_s}\restriction s$. (This is possible by our second inductive hypothesis. Specifically, $\mathcal{N}_k$ satisfied $\mathcal{N}_{k,t} \cong \mathcal{M}_{e_s}\restriction t$ at the most recent stage $t < s$ with $e_t = e_s$ and has not changed since then.)

   Steps 1 and 2 together ensure that $\mathcal{N}_{k,s} \cong \mathcal{M}_{e_s}\restriction s$, no matter whether $\mathcal{N}_k$ was previously a follower of $\mathcal{M}_e$ or not.

3. (**Some $\mathcal{N}_{k',s-1}$ besides $\mathcal{M}_{e_s}$'s follower is isomorphic to $\mathcal{M}_{e_s}\restriction s$.**)

   Suppose there is some $k' \neq k$ such that $\mathcal{N}_{k',s-1} \cong \mathcal{M}_{e_s}\restriction s$. In this case, we release this k' from being the follower of any $\mathcal{M}_{e'}$ for which it was a follower at stage $(s-1)$, and (whether it was released here or previously) we say that k' has been *displaced by* $\mathcal{M}_{e_s}$ at this stage. Whether or not $\mathcal{N}_{k'}$ is a follower at stage $(s-1)$, our $\boldsymbol{d}$-oracle allows us to find the greatest stage $t \leq s$ for which there exists an e with $\mathcal{N}_{k',s-1} \cong \mathcal{M}_e\restriction t$, and also to find this e, by our induction hypothesis and Assumption (2b). By Assumption (1b), there is some stage $t' > s$ and some $\mathcal{M}_j$, both of which we can find with our $\boldsymbol{d}$-oracle, such that:

   - $\mathcal{N}_{k',s-1} \cong \mathcal{M}_e\restriction t$ is a proper substructure of $\mathcal{M}_j\restriction t'$ and
   - for all $i < s$, $\mathcal{M}_j\restriction t'$ is not isomorphic to $\mathcal{M}_i\restriction s$.

   We add elements to $\mathcal{N}_{k'}$ to make $\mathcal{N}_{k',s} \cong \mathcal{M}_j\restriction t'$.

   (If there were more than one such $k' \neq k$, these instructions for Step 3 would have us repeat the process again for each such k'. In fact, though, this step ensures that all $\mathcal{N}_{k,s}$ are pairwise nonisomorphic, for all k which have been chosen as followers up to this stage. So, by induction on the preceding stages, there will be at most one such k'. The induction continues since Case 3 is the only case that changes followers and Step 3 ensures $\mathcal{N}_{k,s} \not\cong \mathcal{N}_{k',s}$.)

This ends stage s, and the construction is now complete. Also, the inductive hypotheses stated earlier are now clear.

We follow Friedberg's argument to show that the $\boldsymbol{d}$-computable enumeration $\langle \mathcal{N}_i \rangle_{i \in \omega}$ thus produced is in fact a classification of the entire family $\mathcal{K}$ of structures. Clearly it is a $\boldsymbol{d}$-computable enumeration of structures.

Lemma 2.4 *If $\mathcal{M}_e \not\cong \mathcal{M}_i$ for all $i < e$, then there exists some k with $\mathcal{N}_k \cong \mathcal{M}_e$.*

Proof. We follow Friedberg [6, Lemma 3, p. 313]. Fix $c = \max_{i<e} \lim_s c(e, i, s)$, which must be finite. Then no follower $\mathcal{N}_k$ of $\mathcal{M}_e$ with $k > c$ will ever be released, and so there is a stage s_0 after which $\mathcal{M}_e$ never loses a follower. Therefore, if at any stage $s > s_0$ with $e_s = e$ we reach Case 3, then $\mathcal{M}_e$ will thereafter have a follower k which it never loses. From then on, whenever $\mathcal{M}_e \restriction (t+1) \not\cong \mathcal{M}_e \restriction t$, if Case 3 applies at the next stage $s > t$ with $e_s = e$, Step 2 of Case 3 will add elements to $\mathcal{N}_k$ to make $\mathcal{N}_{k,s} \cong M_e \restriction s$ again, whereas no other elements will ever be added to $\mathcal{N}_k$ at any other stage. Thus, if Case 3 occurs infinitely often with $e_s = e$, then $\mathcal{N}_k \cong \mathcal{M}_e$.

If Case 3 occurs only finitely often with $e_s = e$, then Case 2 occurs at infinitely many stages instead. (Case 1 would cause $\mathcal{N}_k$ to be released, which will never happen.) At each such stage, there is some k' with $\mathcal{N}_{k',s-1} \cong \mathcal{M}_e \restriction s$, satisfying one of the disjuncts of Case 2. In particular $\mathcal{N}_{k'}$ is one of the finitely many (by induction) followers of some $\mathcal{M}_i$ with $i < e$, $\mathcal{N}_{k'}$ is not a follower and $k' \leq e$, or $\mathcal{M}_e$ previously displaced $\mathcal{N}_{k'}$ in Case 3. But $\mathcal{M}_e$ only executed Case 3 at finitely many stages, so there are only finitely many k' in all for which any of these conditions could hold. Therefore, one of those k' satisfies $\mathcal{N}_{k',s-1} \cong \mathcal{M}_e \restriction s$ at infinitely many stages s, and therefore $\mathcal{N}_{k'} \cong \mathcal{M}_e$. ∎

Lemma 2.4 and the construction now imply that every $\mathcal{N}_k$ eventually becomes a follower of an $\mathcal{M}_e$, at least temporarily, just as shown in the proof in [6]. We also now imitate Lemmas 4 and 5 from that proof ([6, p. 315]). We say that $\mathcal{N}_k$ is finitely generated if there exists some t such that $\mathcal{N}_{k,t} = \mathcal{N}_k$, i.e. $\mathcal{N}_k = \mathcal{M}_j \restriction s$ for some $j, s \in \omega$.

Lemma 2.5 *If $k \neq k'$ and $\mathcal{N}_k$ and $\mathcal{N}_{k'}$ are both finitely generated, then $\mathcal{N}_k \ncong \mathcal{N}_{k'}$.*

Proof. By assumption, there exist t and t' such that $\mathcal{N}_{k,t} = \mathcal{N}_k$ and $\mathcal{N}_{k',t'} = \mathcal{N}_{k'}$. Moreover, we saw above that each must eventually become a follower, say of $\mathcal{M}_e$ and $\mathcal{M}_{e'}$, respectively. Now consider the first stage s such that $\mathcal{N}_k = \mathcal{N}_{k,s} \cong \mathcal{N}_{k',s} = \mathcal{N}_{k'}$ (and such that k and k' have both become followers by stage s). Either one of k and k' became a follower at this stage, or else the congruence arose because elements were added to one of $\mathcal{N}_k$ or $\mathcal{N}_{k'}$ at this stage. Therefore, we must been in Case 3 at stage s, and will have executed Step 3 at this stage. Without loss of generality assume that $e_s = e$. Then steps 1 and 2 ensured that $\mathcal{N}_{k,s} \cong \mathcal{M}_e \restriction s$. If $\mathcal{N}_{k',s-1} \ncong \mathcal{M}_e \restriction s$, then no elements would have been added to $\mathcal{N}_{k'}$ at stage s, contradicting $\mathcal{N}_{k',s} \cong \mathcal{N}_{k,s}$. Therefore, $\mathcal{N}_{k',s-1} \cong \mathcal{M}_e \restriction s$, so we executed Step 3 for this k', placing new elements in $\mathcal{N}_{k,s}$ so that $\mathcal{N}_{k',s} \cong \mathcal{M}_j \restriction t' \ncong M_e \restriction s$, using the j and t' found at that step. Thus $\mathcal{N}_{k',s} \ncong \mathcal{N}_{k,s}$, contradicting our choice above of the stage s. So in fact $\mathcal{N}_k \ncong \mathcal{N}_{k'}$. ∎

Lemma 2.6 *If $k \neq k'$ and neither $\mathcal{N}_k$ nor $\mathcal{N}_{k'}$ is finitely generated, then $\mathcal{N}_k \ncong \mathcal{N}_{k'}$.*

Proof. Every $\mathcal{N}_k$ eventually becomes a follower of some $\mathcal{M}_e$. If it is later released by $\mathcal{M}_e$, then thereafter it is never again a follower, and may be displaced at most once by each $\mathcal{M}_{e'}$ with $e' < x$ and never by any other $\mathcal{M}_{e'}$. Hence it is modified only finitely often in all, leaving it finitely generated. Thus we may assume that neither of $\mathcal{N}_k$ and $\mathcal{N}_{k'}$ is ever released.

Suppose $\mathcal{N}_k \cong \mathcal{N}_{k'}$, and say that they are followers of $\mathcal{M}_e$ and $\mathcal{M}_{e'}$, respectively. Without loss of generality, take $e < e'$. ($\mathcal{M}_e$ can have at most one follower which it never releases, so with $k \neq k'$, we have $e \neq e'$.) Moreover, in order not to be finitely generated, $\mathcal{N}_k$ must undergo Step 2 in Case 3 infinitely often, as must $\mathcal{N}_{k'}$, and therefore $\mathcal{M}_e \cong \mathcal{N}_k \cong \mathcal{N}_{k'} \cong \mathcal{M}_{e'}$. But then $c(e, e', s) \to \infty$ as $s \to \infty$, so there must exist a stage s with $e_s = e'$ at which $c(e, e', s) \geq k'$, and at this stage Case 1 will cause $\mathcal{N}_{k'}$ to be released as a follower of $\mathcal{M}_{e'}$, yielding a contradiction. ∎

Of course, $\mathcal{N}_k \ncong \mathcal{N}_{k'}$ whenever just one of $\mathcal{N}_k$ and $\mathcal{N}_{k'}$ is finitely generated, and so the two preceding lemmas show $\langle \mathcal{N}_k \rangle_{k \in \omega}$ to be one-to-one up to isomorphism. Lemma 2.4 then shows it to be a $\boldsymbol{d}$-computable classification of $\mathcal{K}$ by $\boldsymbol{d}$-computable indices. ∎

To see that Theorem 2.3 applies to the family $\mathcal{K}$ of all computable algebraic field extensions of the prime field Q of characteristic p, we simply use the facts already stated about such fields. Lemma 2.1 gives a computable enumeration of $\mathcal{K}$. Every subfield of a field in $\mathcal{K}$ is also in $\mathcal{K}$, so Assumption (1a) holds. For Assumption (1b), given the finitely generated fields $\mathcal{M}_k\restriction s$ for all $k < s$, fix some prime number $d \neq p$ which is greater than the dimension of each of these fields over Q, and adjoin a d-th root of unity to $\mathcal{M}_i\restriction s$ to get a computably presentable, finitely generated subfield of $\overline{Q}$. Some $\mathcal{M}_j$ in our enumeration of fields must be isomorphic to this subfield, and some t satisfies $\mathcal{M}_j\restriction t = \mathcal{M}_j$ (in fact, j and t can be found effectively), but by the choice of d, we know that $\mathcal{M}_k\restriction s \ncong \mathcal{M}_j\restriction t$. Finally, the assumptions (2a) and (2b) are both standard for algebraic fields. Lemma 2.2 establishes (2a). For (2b), we appeal to Kronecker's Theorem, from [12], as given in [13, Theorem 2.2], for example: it states that we have splitting algorithms for every finitely generated subfield F of $\overline{Q}$, uniformly in the generators of the subfield. This means that, given any $x \in \overline{Q}$, one can find the minimal polynomial $f(X)$ of x over Q and factor $f(X)$ in $F[X]$ effectively; then f has a root in F if and only if at least one of its factors in $F[X]$ is linear. Given F_i and F_j, we can find a primitive generator x for F_i and execute this process. Now F_i embeds into F_j if and only if the minimal polynomial of x over Q has a root in F_j, so the splitting algorithm for F_j tells us whether F_i embeds into F_j. Moreover, $F_i \cong F_j$ if and only if each embeds into the other, so we have a decision procedure for deciding isomorphism as well. This is all that is required by Theorem 2.3, so we have proven the existence of a computable classification of all computable algebraic fields of any fixed charateristic. Finally, our proof is uniform in the characteristic, and hence also yields a computable classification of all computable algebraic fields, as claimed in Theorem 1.9.

Theorem 2.3 works well for families of structures, such as fields of a given characteristic, which have a prime model. The prime model is analogous to the empty set in the original Friedberg construction. Since our theorem requires that every $\mathcal{M}_i\restriction 0$ lie in the family $\mathcal{K}$, however, it is awkward to apply it to families with no prime model. One solution is to consider the empty structure as an element of such a family. In general, though, this difficulty can be avoided by a slight modification to the proof of the theorem.

Corollary 2.7 *Let $\boldsymbol{d}$, $\mathcal{K}$, and $\langle\mathcal{M}_i\rangle_{i\in\omega}$ satisfy all the hypotheses of Theorem 2.3, except that in Assumption (1a), we only require that each $\mathcal{M}_i\restriction(s+1)$ lie in $\mathcal{K}$. Then the conclusion still holds: there exists a $\boldsymbol{d}$-computable classi-*

fication of $\mathcal{K}$ by $\boldsymbol{d}$-computable indices.

Proof. We simply regard every $\mathcal{M}_i\!\upharpoonright\!0$ as empty, and likewise regard $\mathcal{N}_{k,s}$ as empty for every stage s at which $\mathcal{N}_k$ has not yet been chosen as a follower of any $\mathcal{M}_e$. Every $\mathcal{N}_k$ is eventually chosen as a follower, and when it is (in Case 3, at some stage $s > 0$), the construction sets $\mathcal{N}_{k,s} \cong \mathcal{M}_{e_s}\!\upharpoonright\! s$, which lies in $\mathcal{K}$. Therefore, no $\mathcal{N}_k$ winds up empty, and the rest of the proof proceeds exactly as for Theorem 2.3. ∎

3 Torsion-Free Abelian Groups of Rank 1

The construction of Theorem 2.3 does *not* apply to the family $\mathcal{T}$ of computable finite-branching trees, nor to the family $\mathbf{TFAb}_1$ of torsion-free abelian groups of rank 1, and its failure to do so demonstrates the sharpness of the conditions given in the theorem. For the trees, we will see in Lemma 4.1 that there is no computable enumeration of the computable finite-branching trees (analogous to $\{F_n : n \in \omega\}$ above for fields), so there is certainly no computable classification, even though the other hypotheses of Theorem 2.3 hold. (In particular, the isomorphism problem is exactly the same as for algebraic fields.) For $\mathbf{TFAb}_1$, Proposition 3.1 gives a computable enumeration, yet Corollary 3.3 below implies that there is no computable classification of $\mathbf{TFAb}_1$. Here Theorem 2.3 does not apply since the isomorphism problem is no longer Π^0_2 (see Lemma 3.2): we do not have any nice way of comparing two such groups and guessing whether they are isomorphic. In this section we prove these results for $\mathbf{TFAb}_1$.

Proposition 3.1 *There is a computable enumeration of the family $\boldsymbol{TFAb}_1$ of all computable torsion-free abelian groups of rank 1.*

Proof. The short version of this proof simply fixes a computable presentation of the additive group $(\mathbb{Q}, 0, +)$ and lists out the subgroups generated by each c.e. subset W_e of its domain, similar to the argument in Lemma 2.1 for algebraic fields. Here we give a more formal version.

For each e, we define a total function $f_e : \omega \to (\omega + 1)$ by taking $f_e(n) = \lim_s g_e(n, s)$, with g_e nondecreasing in s defined below. With e and n fixed, let $t_0 = 0$, and set

$$t_{s+1} = \begin{cases} t_s, & \text{if } \varphi_{e,s}(n, t_s)\!\uparrow \\ 1 + t_s, & \text{if } \varphi_{e,s}(n, t_s)\!\downarrow, \end{cases}$$

and then define $g_e(n,s)$ (for this single n and for all s) by

$$g_e(n,s) = \begin{cases} 0, & \text{if } s = 0; \\ g_e(n, s-1), & \text{if } \varphi_{e,s}(n, t_s)\uparrow \\ \max(g_e(n, s-1), \varphi_e(n, t_s)), & \text{if } \varphi_{e,s}(n, t_s)\downarrow . \end{cases}$$

Thus g_e is computable and nondecreasing in s. Its limit f_e is the tuple from $(\omega+1)^\omega$ defining the e-th group in our enumeration as follows. We let $G_{e,s}$ be the additive subgroup of $\mathbb{Q}$ generated by the elements $p_n^{-g_e(n,s)}$, where p_n is the n-th prime, for every $n \leq s$. Clearly this gives a computable index for the group G_e, effectively. Moreover, every G_e lies in $\mathbf{TFAb}_1$, and every group G in $\mathbf{TFAb}_1$ is isomorphic to some G_e, since we can take e to be the index of the function $\varphi_e(n,s)$ which measures the number of times that one fixed element $z \in G$ is divisible by p_n as of the s-th stage in the enumeration of G. ∎

Lemma 3.2 *The isomorphism problem E for $\boldsymbol{TFAb}_1$ is Σ^0_3-complete under m-reducibility.*

Proof. The condition for isomorphism of $G, H \in \mathbf{TFAb}_1$ is that there exist nonzero elements $x \in G$ and $y \in H$ such that, for every prime power p^n, x is divisible by p^n in G if and only if y is divisible by p^n in H. So E is Σ^0_3.

We build an m-reduction from the set $\mathbf{Cof}$ to E. Given any $e \in \omega$, we compute indices i, j for torsion-free abelian groups G_i and G_j corresponding to this e as follows. $G_{i,0}$ is the additive group $\mathbb{Z}$, with the element 1 of this $\mathbb{Z}$ named x in $G_{i,0}$. $G_{j,0}$ is a subgroup of the additive group $\mathbb{Q}$, with the element 1 designated as y, but $G_{j,0}$ is generated by all elements in $\mathbb{Q}$ of the form $\frac{1}{p}$ with p prime. Thus y is divisible exactly once in $G_{j,0}$ by each prime, whereas x is not divisible in $G_{i,0}$ by any prime.

Ar stage $s+1$, for each element n (if any) in $W_{e,s+1} - W_{e,s}$, we make both x (in $G_{i,s+1}$) and y (in $G_{j,s+1}$) infinitely divisible by the n-th prime p_n, by adding to each of those two groups all elements of $\mathbb{Q}$ of the form $(p_n)^{-k}$ with $k > 0$, and closing under addition and negation. (Of course, $(p_n)^{-1}$ was already in $G_{j,s}$.) This is the entire construction.

Now, if $e \in \mathbf{Cof}$, then for all but finitely many primes p, both x and y will be divisible infinitely often by p, in G_i and G_j respectively, and thus $G_i \cong G_j$. However, if $e \notin \mathbf{Cof}$, then for infinitely many primes p, y will be divisible by p in G_j, yet x will not be divisible by p in G_i, and so in this case $G_i \not\cong G_j$. Thus we have an m-reduction from $\mathbf{Cof}$ to the isomorphism problem E on $\mathbf{TFAb}_1$. ∎

For the next Corollary, it may be useful to review Definition 1.7.

Corollary 3.3 *There is no* $\mathbf{0}''$*-computable classification of* $\boldsymbol{TFAb}_1$ *by computable indices.*

Proof. To prove this, we establish a strong version of the second part of Proposition 1.5, stating that, for each $n \in \omega$, no family $\mathcal{K}$ of computable structures, closed under isomorphism, whose isomorphism problem is Σ^0_n-complete and whose index set is Δ^0_n has a $\mathbf{0}^{(n-1)}$-computable classification by computable indices.

Suppose that f were a $\mathbf{0}^{(n-1)}$-computable total function classifying $\mathcal{K}$ by computable indices. Then, with a $\mathbf{0}^{(n-1)}$-oracle, we could decide the isomorphism problem E for $\mathcal{K}$ as follows. Given indices i and j, first use the oracle to check whether they are both indices of elements of $\mathcal{K}$. If so, then use the oracle to enumerate the Σ^0_n set E until we find numbers a and b such that $(i, f(a)) \in E$ and $(j, f(b)) \in E$. This must happen, because the image of the classification f contains an index for a computable copy of each computable torsion-free abelian group. However, because the image contains only one such index for each such group, we know that $(f(a), f(b)) \in E$ iff $a = b$. Therefore, $(i, j) \in E$ iff $a = b$, Hence, E would be a Δ^0_n set, a contradiction.

Since the index set of $\mathbf{TFAb}_1$ is Π^0_2 and hence Δ^0_3, Lemma 3.2 then establishes the Corollary. ∎

Lemma 3.4 *There is a* $\mathbf{0}'''$*-computable classification of* $\boldsymbol{TFAb}_1$ *by computable indices.*

Proof. It is simple to construct such a $\mathbf{0}'''$-computable classification g. A $\mathbf{0}'''$-oracle can decide both the index set I for $\mathbf{TFAb}_1$ and its isomorphism problem E, since these are Π^0_2 and Σ^0_3, respectively. So let $g(0)$ be the least element of I, and for each n, let $g(n+1)$ be the least element $j \in I$ with $j > g(n)$ and $(\forall i < j)\ (i, j) \notin E$. This suffices. (Indeed, since g is an increasing function, its image is also $\mathbf{0}'''$-decidable.) ∎

Lemma 3.4 and Corollary 3.3 together prove Theorem 1.10. We are left with the following natural question, which is answered elsewhere in this article for all other structures we consider, but remains open for $\mathbf{TFAb}_1$.

Question 3.5 *Is there a* $\mathbf{0}''$*-classification of* $\boldsymbol{TFAb}_1$ *by* $\mathbf{0}''$*-indices? If so, is there a* $\mathbf{0}'$*-classification by* $\mathbf{0}'$*-indices?*

We draw attention to the contrast between Theorem 1.9 and Theorem 1.10. Algebraic fields and rank-1 torsion-free abelian groups are usually regarded as highly similar families of structures. In each case, every element x of a computable model of the structure can be identified effectively up to finitely many possibilities: in fields, finding the minimal polynomial of x over the prime subfield accomplishes this, while in groups, having fixed a single non-identity element z, one finds a nontrivial relation on z and x, expressed as $x = qz$ for some $q \in \mathbb{Q}$. Such a relation must exist, since the group has rank 1, and once it is found, x is known to be the unique element satisfying it, since the group is isomorphic to an additive subgroup of $\mathbb{Q}$. One might suspect that therefore the groups would be more amenable to classification than the fields, at least given finitely much information (namely the parameter z). Theorems 1.9 and 1.10 reverse this intuition.

Moreover, in computable structure theory, it is known that these two families have exactly the same possible spectra. Recall the relevant definition.

Definition 3.6 For a countable structure $\mathcal{A}$, the *spectrum* of $\mathcal{A}$ is the set of all Turing degrees of structures isomorphic to $\mathcal{A}$:

$$\mathrm{Spec}(\mathcal{A}) = \{\deg(\mathcal{B}) : \mathcal{B} \cong \mathcal{A} \ \& \ \mathrm{dom}(\mathcal{B}) = \omega\}.$$

(We consider only structures $\mathcal{B}$ with domain ω, so the degree of $\mathcal{B}$ is always a well-defined concept.)

The following result was proven for $\mathbf{TFAb}_1$ by Coles, Downey, and Slaman in [4], and for algebraic fields by Frolov, Kalimullin, and Miller in [7].

Theorem 3.7 *For every set $\mathcal{U}$ of Turing degrees, the following are equivalent.*

- *$\mathcal{U}$ is the spectrum of some infinite algebraic field.*
- *$\mathcal{U}$ is the spectrum of some torsion-free abelian group of rank* 1*.*
- *There exists a set $U \subseteq \omega$ for which*

$$\mathcal{U} = \{\boldsymbol{d} : U \textit{ is } \boldsymbol{d}\textit{-computably enumerable}\}.$$

The third item says that $\mathcal{U}$ is best viewed as an upper cone of e-degrees above some specific set U, being defined by the ability to enumerate the set

U. (For an algebraic field F with prime field Q, we can take U to be the set of polynomials in $Q[X]$ with roots in F. For an (additive) group G, fix any non-identity element $z \in G$, and let

$$U = \{\langle n,k\rangle \in \omega^2 : (\exists x \in G)\ p_n^k \cdot x = z\}.)$$

Again, this reflects the nature of these two families of structures: members of each can be described by existential conditions. In §4, we consider another family of this type: the family $\mathcal{T}$ of all computable finite-branching trees under the predecessor function.

However, despite the similarities between the families of algebraic fields and torsion-free abelian groups, their classification problems (for computable structures) turned out to be of significantly different complexity: we found a computable classification of all computable algebraic fields, whereas, using computable indices, $\mathbf{TFAb}_1$ has only a $\mathbf{0}'''$-computable classification.

4 Finite-Branching Trees

Recall that $\mathcal{T}$ is the class of all computable finite-branching trees, under the function P which maps each node in a tree to its immediate predecessor. (By convention, P maps the root of a tree to itself.) We start with a straightforward lemma, subsequently superseded by Proposition 4.3.

Lemma 4.1 *There is no $\mathbf{0}'$-computable enumeration of $\mathcal{T}$ by computable indices.*

Proof. Suppose f were such an enumeration, with computable approximation g, so that $f(x) = \lim_s g(x,s)$ for every x. Since that language only includes the unary function P, we can think of $\varphi_{f(x)}$ as being the computable function P on ω.

We build a computable tree T such that, for all x, we have $T \not\cong T_{f(x)}$. At stage 0, we start with a "spine" for T: a single node at each level, forming an infinite path. At stage s, for each $e \leq s$, check how many numbers $y \leq s$ have $\varphi^e_{g(e,s),s}(y) = \varphi^{e+1}_{g(e,s),s}(y)$, where $\varphi^n_{g(e,s),s}$ means the composition of $\varphi_{g(e,s),s}$ with itself n times. (Since the root is defined to be its own predecessor, this determines how many nodes $y \leq s$ in this tree lie at levels less than or equal to e. The condition says that the e-th predecessor equals the $(e+1)$-st predecessor, i.e. that the e-th predecessor is the root.) If k many such y exist, add $k+1$ many nodes to level e of T, all with their immediate predecessors on the spine of T. (If T already had more than k nodes at level e, then do nothing.) Eventually $g(x,s)$ converges to $f(x)$, and from that stage onwards, $\varphi_{f(x)} = \varphi_{g(x,s)}$ computes the predecessor function of a single finite-branching tree. Therefore, once all the nodes at levels $l \leq e$ in this tree have appeared, T never again adds any new nodes at its own level e. Thus T is indeed computable and finite-branching but not isomorphic to the tree computed by $\varphi_{f(x)}$. So T was omitted by the enumeration f, a contradiction. ■

As seen in this proof, it is often simplest to view a finite-branching tree just as a tree in which each level has only finitely many nodes. Nevertheless, the usual definition of finite-branching (that each node has only finitely many immediate successors) has the least possible complexity, as we now show.

Lemma 4.2 *The index set I for the family $\mathcal{T}$ is Π^0_3-complete.*

Proof. To see that I is Π^0_3, notice that the partial computable function φ_e is the predecessor function for a finite-branching tree with domain ω if and only if the following all hold.

- φ_e is total.
- There is a unique r for which $\varphi_e(r) = r$.
- For every $x \in \omega$, there exists an l such that $\varphi^l_e(x) = \varphi^{l+1}_e(x)$. (This means that, for the least such l, the l-th predecessor $\varphi^l_e(x)$ of x is the root, so that x lies at the level l of the tree.)
- For every $l \in \omega$, there are only finitely many $x \in \omega$ with $\varphi^l_e(x) = \varphi^{l+1}_e(x)$. (This says that the tree has only finitely many nodes at each level l, which is equivalent to being finite-branching.)

One can also check, using only a $\mathbf{0}''$-oracle, whether φ_e has finite domain and computes a tree on that domain. Therefore the set I (even including indices of finite trees) is Π^0_3.

To show that I is Π^0_3-complete under m-reducibility, we give an m-reduction from the complement of **Cof** to I. Given any index $e \in \omega$, build the computable tree T_e with root 0 as follows. At stage s, let the least fresh element of ω lie at level $s+1$ in T_e, with the least node at level s as its immediate predecessor. Then, if the n-th smallest element of the complement of $W_{e,s}$ lies in the set $W_{e,s+1}$, add a new node to level $n+1$ of T_e, with the least node at level n as its immediate predecessor. This is the entire construction. If $e \in \mathbf{Cof}$, then for $n = |\overline{W_e}|$, the $(n+1)$-st smallest element of $W_{e,s}$ entered $W_{e,s+1}$ at infinitely many stages s, and therefore T_e is infinite-branching, with infinitely many nodes at level $(n+2)$. On the other hand, if $e \notin \mathbf{Cof}$, then for every n, the $(n+1)$-st level of T_e only received a new node at finitely many stages, and so T_e is finite-branching. Thus we have the necessary m-reduction. ■

Proposition 4.3 *There is no* $\mathbf{0}''$*-computable enumeration of all computable finite-branching trees by computable indices.*

Proof. First note that the isomorphism problem E for computable finite-branching trees is Π^0_2, since two finite-branching trees are isomorphic if and only if every finite subtree of each one embeds into the other. (For details, see [17].) Moreover, the same statement holds of any two computable trees under predecessor, provided only that at least one of them is finite-branching. Suppose S is a finite-branching tree and T is an infinite-branching tree. Then we can choose the least infinite-branching node $x \in T$, say at level l, and consider the finite subtree consisting of x, its predecessors, and $(a+1)$ of its immediate successors, where a is the number of nodes at level $(l+1)$ in the finite-branching tree S. Clearly this finite subtree of T cannot embed into S (recalling that an embedding must map the root to the root), and so the Π^0_2 condition fails for this pair (S,T). (The Π^0_2 condition can hold for non-isomorphic S and T when both are infinite-branching.)

With this information we can show that the existence of such an enumeration f would force the index set I of the family of computable finite-branching trees to be Σ^0_3. Indeed, an index e would lie in I if and only if φ_e computes a tree T_e under predecessor, either with domain ω or with finite domain (all of which is $\mathbf{0}''$-decidable), and there exists some $n \in \omega$ such that every finite subtree of each of T_e and $T_{f(n)}$ embeds into the other (which is Σ^0_3, including the quantifier $(\exists n)$). Indeed, since $T_{f(n)}$ is known to be finite-branching, the preceding paragraph shows that $T_e \cong T_{f(n)}$ if and only if the Π^0_2 condition on embedding of finite subtrees holds; conversely, if T_e really is finite-branching, then it must be isomorphic to some $T_{f(n)}$. Thus I would be Σ^0_3, contrary to Lemma 4.2. ■

Theorem 1.11 follows from Proposition 4.3 along with the next results.

Proposition 4.4 *There exists a* $\mathbf{0}'''$*-computable classification of all computable finite-branching trees by computable indices.*

Proof. The $\mathbf{0}'''$-classification f is readily given: $f(n)$ is the least $m > f(n-1)$ (or the least $m \geq 0$, if $n = 0$) such that m lies in the index set for computable finite-branching trees and, for all $k < n$, the tree T computed by φ_m is not isomorphic to that computed by $\varphi_{f(k)}$. Lemma 4.2 shows that the first part of this is a Π^0_3 condition, hence decidable by our $\mathbf{0}'''$-oracle, and the isomorphism problem for these trees is Π^0_2, as discussed in Proposition 4.3. In fact, the image of this classification function f is Δ^0_4, since f itself is strictly increasing. ■

Finally, we apply Friedberg's method (as adapted in Theorem 2.3) to give a simpler classification of the finite-branching computable trees: this classification requires only a $\mathbf{0}'$-oracle, but uses $\mathbf{0}'$-computable indices.

Proposition 4.5 *There exists a* $\mathbf{0}'$*-computable classification of the family* $\mathcal{T}$ *of all computably presentable finite-branching trees by* $\mathbf{0}'$*-computable indices.*

Proof. In order to fulfill the hypotheses of Theorem 2.3, we first give a $\mathbf{0}'$-computable enumeration g of $\mathcal{T}$ by $\mathbf{0}'$-computable indices. For each e, let $g(e)$ be an index for the $\mathbf{0}'$-computable tree T_e defined as follows, with root 0. Use the oracle to ask whether φ_e has a fixed point r, and, if so, whether it is unique (i.e. ask whether $\forall x \forall s(\varphi_{e,s}(x)\!\downarrow = x \implies x = r)$). If not, then T_e consists just of its root 0. If so, then, using the oracle, check that $\varphi_e(0)\!\downarrow$, and search for an m such that no element greater than m has r as a predecessor (i.e. ask whether $(\forall s \forall x > m)(\neg\varphi_{e,s}(x)\!\downarrow = r)$). If $\varphi_e(0)\!\uparrow$, or if we never find such an m, then again T_e will consist only of the root. Otherwise, having found m, we add one node at level one to T_e for each $x \leq m$ with $x \neq r$ and $\varphi_e(x)\!\downarrow = r$ (which our oracle can check). Thus we have a one-to-one correspondence between the nodes at level 1 in T_e and those at level 1 in the tree S_e (if any) computed by φ_e, provided that S_e is finite-branching at its root. This completes level 1 in T_e.

Next, we repeat the process at level 1. Provided $\varphi_e(1)\!\downarrow$, we assign to each individual node at level 1 in T_e one of the nodes at level 1 in S_e and repeat this process with that node in place of the root. We then repeat this process (unless it terminates) at each level. Of course, if φ_e computes a tree S_e which is not finite-branching, then our T_e will be a finite tree. (Also, if

φ_e is not total or fails to compute a tree, then again our T_e will be a finite tree.) However, if S_e is a finite-branching tree with domain ω, then T_e not only will also be a finite-branching tree, but will in fact be isomorphic to S_e.

Finally, notice that if S_e is a tree with domain $\{0, \ldots, n\}$, then this process will build T_e isomorphic to S_e, stopping when it finds that $\varphi_e(n+1) \uparrow$. Therefore, every finite tree appears on our list. So the set $\{T_e\}_{e \in \omega}$, given by a $\mathbf{0}'$-computable list of indices for $\mathbf{0}'$-computable trees, includes a presentation of every computable finite-branching tree, yet includes only trees which are isomorphic to computable finite-branching trees.

The remaining assumptions of Theorem 2.3 are readily seen to hold. The restriction $T_i \restriction s$ of any T_i in the enumeration is actually already downward closed and is an element of $\mathcal{T}$. Given s, let h be the maximum of the heights of all trees $T_k \restriction s$ with $k < s$. Given i and s, find some j and t such that $T_j \restriction t$ contains a node at height $(h+1)$ and $T_i \restriction s$ embeds into $T_j \restriction t$. This j and t establish Assumption (1b) of Theorem 2.3. Assumption (2a) is already known to hold of finite-branching trees (see e.g. [17]). In Assumption (2b), the trees $T_i \restriction t$ and $T_j \restriction s$ are both finite (and the size of the domain of each is $\mathbf{0}'$-computable). So it is simple to check using $\mathbf{0}'$ whether they are isomorphic and whether the first embeds into the second. Therefore, Theorem 2.3 yields a $\mathbf{0}'$-computable classification of the family $\mathcal{T}$ by $\mathbf{0}'$-computable indices. ■

5 Computable Equivalence Structures

An *equivalence structure* is simply an equivalence relation E on a given domain D. For an equivalence structure to be computable, we require D to be an initial segment of ω, and E a computable subset of $\omega \times \omega$. Notice that this definition specifically allows finite equivalence structures. We normally write $[x]_E$ for the E-equivalence class containing the element x of the domain.

The principal distinction among equivalence structures arises from the number of infinite equivalence classes defined by the relation E. We denote the family of those computable equivalence structures containing exactly n equvalence classes by $\mathcal{E}_n$ (for each $n \in \omega$). The finite equivalence structures (with domain an initial segment of ω) are all included in $\mathcal{E}_0$, and we specifically include the empty structure in $\mathcal{E}_0$.

5.1 Classifications by computable indices

Recall Theorem 1.3 of Goncharov and Knight, which states that there exists a computable classification of the family $\mathcal{E}_\infty$ of computable equivalence structures with infinitely many infinite equivalence classes. Our approach is to study the possibility of classifying $\mathcal{E}_0$. A classification of $\mathcal{E}_0$ would yield a classification of $\mathcal{E}_n$, for each $n < \omega$, because each structure in $\mathcal{E}_n$ is the disjoint union (in a unique way) of a structure containing n infinite classes (and nothing else) with a structure containing no infinite equivalence classes. Putting this together with Theorem 1.3 would yield a classification of the family $\mathcal{E}$ of all computable equivalence structures.

Lemma 5.1 *The isomorphism problem for the family $\mathcal{E}_0$ of computable equivalence structures with no infinite classes is Π^0_3-complete.*

Proof. To see that the isomorphism problem is Π^0_3, notice that the equivalence relations E_i and E_j on ω computed by φ_i and φ_j are isomorphic if and only if i and j lie in the index set for $\mathcal{E}_0$ (which is readily seen to be Π^0_3) and, for every n and k, each one has at least n classes of size exactly k if and only if the other does. For a given element to lie in a class of size exactly k is $\mathbf{0}'$-decidable, since there are no infinite classes. So the given condition is that, for every k and all pairwise-E_i-inequivalent $x_1, \ldots, x_n$, there exist pairwise-E_j-inequivalent $y_1, \ldots, y_n$ such that

$$(\text{every } [x_m]_{E_i} \text{ has size } k) \implies (\text{every } [y_m]_{E_j} \text{ has size } k),$$

along with the same statement with i and j reversed. This is Π^0_3.

For each input e, we build a pair of computable equivalence structures E_e and F_e, uniformly in e. Neither E_e nor F_e will have any infinite equivalence classes, and E_e and F_e will be isomorphic iff $e \notin \mathbf{Cof}$. This will prove the lemma.

At stage 0, each of E_e and F_e has one class of each finite size. At stage $s+1$, if $W_{e,s+1} = W_{e,s}$, we change nothing. If some (single) element x has entered W_e at stage $s+1$, fix the $n \geq 0$ such that the complement $\overline{W}_{e,s}$ contained exactly n elements $< x$. By induction, E_e and F_e each contain exactly one class of size $2n+1$, and contain the same number of classes of size $2n+2$. We add $2n+2$ new elements to each structure. In E_e, these new elements form a new class of size $2n+2$. In F_e, one of these elements is added to the unique class of size $2n+1$, forming a new class of size $2n+2$, and the remaining new elements form a new class of size $2n+1$. This is the entire construction.

Now if $e \in \mathbf{Cof}$, then for some (minimal) n, there are infinitely many stages $s+1$ at which the $(n+1)$-th smallest element of $\overline{W}_{e,s}$ enters $W_{e,s+1}$. Consequently, F_e has no class of size $2n+1$, since every F_e-class with $2n+1$ elements eventually receives another element. However, the original E_e-class of size $2n+1$ never receives any more elements, and so $E_e \ncong F_e$. Conversely, if $e \notin \mathbf{Cof}$, then for every n there is a stage s such that none of the $(n+1)$ smallest elements of $\overline{W}_{e,s}$ ever enters W_e, and so the F_e-class with $2n+1$ elements as of stage s never acquires any more elements. Thus F_e has exactly one class with $2n+1$ elements, as does E_e. Moreover, as of stage s, they have the same number of classes of size $2n+2$, and those classes never change at any subsequent stage. This holds for every n, so $E_e \cong F_e$ in this case, proving the lemma. ∎

In this proof we remarked that the index set for $\mathcal{E}_0$ is Π^0_3. In fact, it is complete at this level.

Lemma 5.2 *The index set for the family $\mathcal{E}_0$ of computable equivalence structures with no infinite classes is Π^0_3-complete.*

Proof. Fix any $e \in \omega$. As with Lemma 5.1, we consider the "markers" on the complement of W_e at each stage as we build the equivalence relation E_e. At each stage s, we add one new element x_s to E_e, in a new E_e-class. Also, if the n-th marker moved at stage s (and n is minimal with this property), then we add another new element to E_e, in the E_e-class of x_n. (This assumes that W_e is enumerated so that, at stage s, no $x \geq s$ enters W_e; thus x_n must be defined at this stage.) This is the entire construction.

Now if $e \in \mathbf{Cof}$, fix the least n such that the n-th marker moves at infinitely many stages. The x_n lies in an infinite E_e-class, and so the index of E_e is not in the index set for computable equivalence relations with no infinite equivalence classes. On the other hand, if $e \notin \mathbf{Cof}$, then the index of E_e does lie in this index set, since for every n there is a stage after which the n-th marker never moves again, and so the equivalence class of each x_n is finite for every n. Thus we have an m-reduction from $\mathbf{Cof}$ to the complement of the index set. ∎

Corollary 5.3 *There exists a $\mathbf{0}'''$-computable classification of the computable equivalence structures with no infinite classes, by computable indices.*

Proof. With a $\mathbf{0}'''$-oracle, for each n, we may compute the least number $f(n)$ such that:

- $f(n) > f(n-1)$ or $n = 0$; and
- $f(n)$ lies in the index set from Lemma 5.2; and
- for all $m < n$, $\langle f(m), f(n) \rangle$ does not lie in the isomorphism problem from Lemma 5.1.

Not only is this f the required classification, but it is also strictly increasing, so its image is $\mathbf{0}'''$-computable. ■

Corollary 5.4 *There exists a* $\mathbf{0}'''$*-computable classification of all computable equivalence structures by computable indices.*

Proof. Let g_0 be the computable function from Theorem 1.3, classifying all computable equivalence relations with infinitely many infinite classes. Let f be the classification given in Corollary 5.3, and, for each $n > 0$, define $g_n(x)$ to be the index of a computable equivalence structure which, on the even numbers, builds an isomorphic copy of the equivalence structure given by the index $f(x)$, and partitions the odd numbers into exactly $(n-1)$ equivalence classes, all infinite. (For the special case $n = 1$, $g_1(x)$ uses all of ω, not just the even numbers, to build the equivalence relation given by $f(x)$.) Finally, we define $g(\langle n, x \rangle) = g_n(x)$, giving a $\mathbf{0}'''$-computable classification g of all computable equivalence structures by computable indices. ■

The following theorem is in fact a consequence of the more general Theorem 5.6 below. We include it with proof here to introduce the natural method of diagonalizing against computable enumerations of families of equivalence structures.

Theorem 5.5 *There is no computable enumeration (let alone any computable classification) of the family* $\mathcal{E}_0$ *of computable equivalence structures with no infinite classes.*

Proof. Suppose that $E_0, E_1, \ldots$ were the structures in such an enumeration. To derive a contradiction, we show how to build a new structure E in $\mathcal{E}_0$ that is not isomorphic to any E_n. This E will satisfy requirements:

$\mathcal{R}_e$: If E_e contains arbitrarily large classes, then for some k_e, E_e contains a class of size k_e and E does not.

E itself will contain arbitrarily large classes, but no infinite classes, so satisfaction of these requirements will prove the theorem: the purported enumeration of $\mathcal{E}_0$ does not include E.

First, set $m_0 = y_0 = 0$, and let $k_{0,s} = |\{x \leq s : \langle y_0, x\rangle \in E_0\}|$. So the sequence $k_{0,s}$ is nondecreasing with limit k_0, the size of $[0]_{E_0}$, which by hypothesis is finite. We build an equivalence class $[y_0]_E$ in E, starting with two elements (i.e. we place $\langle 0, 1\rangle \in E$). At each stage s for which $k_{0,s+1} \neq k_{0,s}$, we add the least available fresh element to the class $[y_0]_E$. Thus the class $[y_0]_E$ will contain exactly k_0+1 elements of ω, and (by the rest of the construction) every other class in E will contain more than k_0 elements, so k_0 will witness that $\mathcal{R}_0$ holds.

Meanwhile, we employ a similar strategy in regard to $\mathcal{R}_1$. Starting at stage 1, we fix the least element y_1 with $\langle y_0, y_1\rangle \notin E$. We set $k_{1,1} = k_{0,1} + 2$, put $k_{1,1} + 1$ elements (including y_1) in the E-class of y_1 at this stage, leave $m_{1,1}$ undefined, and follow the ensuing instructions at subsequent stages $s+1$.

1. If $k_{0,s+1} > k_{0,s}$, then the strategy for $\mathcal{R}_1$ is *injured* at this stage. The element y_1 remains fixed, but $m_{1,s+1}$ becomes undefined. We set $k_{1,s+1} = \max(k_{1,s}, k_{0,s+1} + 2)$, and add fresh elements to $[y_1]_E$ if needed to ensure that this class still contains at least $k_{1,s+1} + 1$ elements.

2. If $k_{0,s+1} = k_{0,s}$ but $m_{1,s}$ is undefined, we check whether E_1 has an equivalence class containing at least $k_{1,s}$ elements of $\{0, 1, \ldots, s + 1\}$. If not, we change nothing. If so, then we set $k_{1,s+1}$ to be the number of $x \leq s + 1$ in this class, designate the least element of this class as $m_{1,s+1}$, and add fresh elements to the class of y_1 in E until the class $[y_1]_E$ contains exactly $k_{1,s+1} + 1$ elements.

3. Otherwise $m_{1,s}$ is defined and $k_{0,s+1} = k_{0,s}$. We set $m_{1,s+1} = m_{1,s}$ and $k_{1,s+1} = |\{x \leq s + 1 : \langle x, m_{1,s+1}\rangle \in E_1\}|$. If $k_{1,s+1} = k_{1,s}$, we do nothing, since by induction the class $[y_1]_E$ contains $k_{1,s+1} + 1$ elements. If $k_{1,s+1} > k_{1,s}$, then we add the least available fresh elements to the class $[y_1]_E$ in E until this class contains $k_{1,s+1} + 1$ elements. (Only one new element should be required.)

Of course, the approximation to k_0 only increases at finitely many stages, so either E_1 contains no equivalence class of size greater than $k_0 + 1$ (in which case $\mathcal{R}_1$ is satisfied), or eventually an element $m_1 = \lim_s m_{1,s}$ stabilizes. It lies in a finite class in E_1, by assumption, so also $k_1 = \lim_s k_{1,s}$ stabilizes at the size of the E_1-class of m_1. But $[y_0]_E$ contained only k_0+1 elements, and $k_0+1 < m_1$ since (after the last injury) $m_{1,s}$ was chosen so that $k_{1,s} \geq k_0+2$. Therefore, $[y_0]_E$ does not contain k_1 elements. On the other hand, $[y_1]_E$ contains k_1+1 elements, and the other classes $[y_2]_E, [y_3]_E, \ldots$ will each contain more elements than that. So this k_1 witnesses the satisfaction of $\mathcal{R}_1$.

The strategy for $\mathcal{R}_{e+1}$ is exactly the same, relative to $\mathcal{R}_e$ and E_e, as the above strategy for $\mathcal{R}_1$ relative to $\mathcal{R}_0$ and E_0. Whenever $\mathcal{R}_e$ is injured, so is $\mathcal{R}_{e+1}$. So, by induction on e, every $\mathcal{R}_e$ is satisfied: either E_e contains no class of size at least $k_{e-1}+2$, or else it contains a class of size k_e, in which case the construction ensures that the classes $[y_i]_E$ all contain either fewer than k_e elements (for $i < e$) or more than k_e elements (for $i \geq e$). Furthermore, E contains no classes except the classes $[y_e]_E$ but does contain arbitrarily large classes (since, given any $n \in \omega$, the enumeration $E_0, E_1, \ldots$ contains an equivalence structure E_e having a class of at least n elements, forcing m_e to exist for infinitely many e). This proves the theorem. ■

In their research leading to this article, the authors proved that for each finite nonempty subset $S \subseteq \omega$, there is no computable enumeration of the family $\mathcal{E}_S = \cup_{n\in S}\mathcal{E}_n$. However, instead of that proof, we present the proof of a stronger result, recently established by Harrison-Trainor, Melnikov, Montalbán, and one of us.

Theorem 5.6 (Harrison-Trainor, Melnikov, Miller, Montalbán) *For every nonempty subset $S \subseteq \omega$, there is no computable enumeration of the family $\mathcal{E}_S = \cup_{n\in S}\mathcal{E}_n$ of all computable equivalence structures E_i in which the number of infinite equivalence classes is an element of S. Hence, there is no computable classification of any such $\mathcal{E}_S$.*

The notation here leads to some possibility of confusion. The family $\mathcal{E}_\omega$, with $S = \omega$, is defined here as the family of those computable equivalence structures with only finitely many infinite classes; this family should not be confused with $\mathcal{E}_\infty$, which is precisely its complement in $\mathcal{E}$. Every $\mathcal{E}_S$ in this theorem is disjoint from $\mathcal{E}_\infty$.

Proof. Suppose that $E_0, E_1, E_2, \ldots$ is a computable enumeration of some $\mathcal{E}_S$. We will produce a computable equivalence structure $E \in \mathcal{E}_S$ that is not isomorphic to any of these E_e, thereby proving the theorem. The construction of our E from the given enumeration is uniform, except we fix one number

$a \in S$. Below we will build an E in $\mathcal{E}_0$ which has arbitrarily large finite classes, but also satisfies the following requirements $\mathcal{R}_e$ for every $e \in \omega$:

$\mathcal{R}_e$: If E_e has arbitrarily large finite classes, then there exists some $k \in \omega$ such that E_e has a class of size k and E does not.

Now if $0 \notin S$, then our E does not satisfy our purpose. However, one then builds an E^* which has all the same finite classes as E, but also has a-many infinite classes. This E^* then lies in $\mathcal{E}_a$, hence in $\mathcal{E}_S$, and the requirements $\mathcal{R}_e$ will then show that the given enumeration failed to list any isomorphic copy of E^*, thus proving the theorem.

Our strategy is to start listing the elements of the classes in each E_e. We begin our basic module against E_e when we find the first element $x_{e,1}$ in any of its equivalence classes. It will next require attention if we reach a stage at which the $[x_{e,1}]_{E_e}$ has at least two elements and a new element $x_{e,1}$ has appeared with $\langle x_{e_2}, x_{e,1}\rangle \notin E_e$. After that, it requires attention at the next stage (if any) at which $[x_{e,1}]_{E_e}$ has at least three elements, $[x_{e,2}]_{E_e}$ has at least two, and a new $x_{e,3}$ has appeared that lies in neither of these classes. We continue in the same fashion forever, claiming that $\mathcal{R}_e$ will require attention at only finitely many stages. Indeed, by hypothesis, only finitely many of these E_e-classes are infinite, so eventually some $x_{e,i}$ will be found that lies in a finite class (or else E_e consists of finitely many infinite classes and nothing else, in which case, for some $i \in \omega$, no $x_{e,i}$ ever appears). After that, $\mathcal{R}_e$ will require attention at most once each time the class $[x_{e,i}]_{E_e}$ expands, hence only finitely many more times in total.

Each time $\mathcal{R}_e$ requires attention, E_e has finitely many equivalence classes $[x_{e,1}]_{E_e}, \ldots, [x_{e,n_e}]_{E_e}$ so far, each with finitely many observed elements. We write $k_{e,j,s}$ for the size of $[x_{e,j}]_{E_e} \cap \{0, 1, \ldots, s\}$ at stage s. The construction then ensures, until the next stage (if any) at which $\mathcal{R}_e$ requires attention, that E contains no E-class of any of the sizes $k_{e,0,s}, \ldots, k_{e,n_e,s}$. If $\mathcal{R}_e$ never again requires attention, then one of the classes $[x_{e,i}]_{E_e}$ never again expands, hence contains exactly $k_{e,i,s}$ elements. In this case E_e cannot be isomorphic to E, since E will have no class of any of the sizes $k_{e,j,s}$ with $j \leq n_e$.

To combine the different basic modules, we use a finite-injury procedure. Every requirement $\mathcal{R}_d$ with $d > e$ will be injured at each stage at which $\mathcal{R}_e$ receives attention. After that stage, instead of just waiting for one element $x_{d,1+n_d}$ in a new equivalence class to appear, $\mathcal{R}_d$ will watch for an E_d-class $[x_{d,1+n_d}]_{E_d}$ to appear which has at least $2 + m_d$ elements in it, where $m_d =$

$\max\{k_{e,j} : e < d, j \leq n_e\}$. Meanwhile, E will create a class $[y_d]_E$ which has exactly $1+m$ elements. This class will help show that E contains arbitrarily large finite classes. However, no class in E will wind up with infinitely many elements (because of the finite-injury nature of the argument), and if E_d also has arbitrarily large finite classes, then after the greatest stage at which it is injured, it will eventually produce an $x_{d,0}$ and the diagonalization against E_d will begin.

At stage 0, we set every $n_{e,0} = 0$ and every $m_{e,0} = 2$. We set $y_e = 2e$ in the structure E and make them all E-inequivalent to each other. We leave all other values undefined at this stage. At the start of stage $s+1$, we have numbers $n_{e,s}$ defined at the preceding stage, and for each e with $n_{e,s} > 0$ we have elements $x_{e,1,s}, \ldots, x_{e,n_{e,s},s}$ in E_e and numbers $k_{e,1,s}, \ldots, k_{e,n_{e,s},s}$. We define the *threshold values*: $m_{0,s+1} = 2$ and, for each $e \leq s$,

$$m_{e,s+1} = \max(m_{e,s}, 2 + \max\{k_{d,j,s} : d < e \ \& \ j \leq n_{d,s}\}),$$

with $m_{e,s+1} = 2$ for all $e > s$. For each $e \leq s$, we add $(m_{e,s+1} - m_{e,s})$ new elements to $[y_e]_E$. (The definition of $m_{e,s+1}$ ensures that $m_{e,s+1} \geq m_{e,s}$, and so this step shows that $[y_e]_E$ has size exactly $(m_{e,s+1} - 1)$ at this stage.)

Now we search for the least $e \leq s$ satisfying the following conditions:

- there exists some $x \leq s$ such that $[x]_{E_e} \cap \{0, 1, \ldots, s\}$ contains at least $m_{e,s}$ elements and, for all $j = 1, \ldots, n_{e,s}$, we have $\langle x, x_{e,j,s} \rangle \notin E_e$; and
- for every $j = 1, \ldots, n_{e,s}$, $[x_{e,j,s}]_{E_e} \cap \{0, 1, \ldots, s\}$ contains at least $k_{e,j,s}+1$ elements. (These conditions on j are vacuous if $n_{e,s} = 0$.)

If no such $e \in \omega$ is found, then we do nothing at this stage. Otherwise, for the least such e, requirement $\mathcal{R}_e$ *receives attention*, as follows. We define $n_{e,s+1} = 1 + n_{e,s}$ (writing $n = n_{e,s+1}$ hereafter) and set $x_{e,n,s+1}$ to be the least x witnessing the first condition above. For each $j \leq n$, we let $x_{e,j,s+1} = x_{e,j,s}$ and reset

$$k_{e,j,s+1} = | \ [x_{e,j,s+1}]_{E_e} \cap \{0, \ldots, s\} \ |,$$

noting that (by induction and the choice of $x_{e,n,s+1}$) every $k_{e,j,s+1} \geq m_{e,s+1}$. For every $d > e$, we reset $n_{d,s+1} = 0$, thus injuring $\mathcal{R}_d$. (The threshold value $m_{d,s+1}$ was defined above and is preserved, but all other values associated to $\mathcal{R}_d$ become undefined at stage $s+1$.) The rest of E remains unchanged, and we preserve all values defined for requirements $\mathcal{R}_d$ with $d < e$. This completes stage $s+1$.

The argument that each $\mathcal{R}_e$ receives attention at only finitely many stages proceeds by induction on e. Fixing the least stage s_0 such that no requirement $\mathcal{R}_d$ with $d < e$ receives attention at any stage $t \geq s_0$, we note that $m_{e,s_0} = m_{e,s}$ for all $s > s_0$, and we write m for this permanent threshold value for $\mathcal{R}_e$. Now if E_e has no equivalence classes of size at least m, then $\mathcal{R}_e$ will never receive attention again. If E_e does have such a class, then $x_{e,1,s+1}$ will be defined at the first stage s at which we observe such a class. Thereafter we keep on searching for more such classes and for increases in the sizes of the existing such classes. Notice that, if $x_{e,j,s}$ becomes defined after stage s_0 for some $j \in \omega$, then $x_{e,j,s}$ is never redefined, so we call it $x_{e,j}$. However, if $\mathcal{R}_e$ receives attention at infinitely many stages after s_0, then we would have infinitely many elements $x_{e,1}, x_{e,2}, \ldots$, since a new one is chosen at each such stage. Also, since each existing equivalence class $[x_{e,j}]_{E_e}$ must expand in order for $\mathcal{R}_e$ to receive attention again, every class $[x_{e,j}]_{E_e}$ would be infinite. Therefore, since E_e has only finitely many infinite classes, there must exist a greatest stage s_1 at which $\mathcal{R}_e$ receives attention. This completes the induction.

Hence, for each fixed e, the limit $m_e = \lim_s m_{e,s}$ exists. By construction, $\mathcal{R}_e$ stops receiving attention because either E_e does not have any finite E_e-classes of size at least m_e (hence $\mathcal{R}_e$ stopped receiving attention once representatives of all its infinite classes had been discovered), or else some $x_{e,i}$ was chosen (after the final injury to $\mathcal{R}_e$) whose equivalence class only expanded at finitely many subsequent stages (hence $[x_{e,i}]_{E_e}$ is finite). In the latter case, we (noneffectively) fix the i that is the first to have its E_e-class reach full size. (That is, choose $i \leq n_e$ so that $\max\, [x_{e,i}]_{E_e}$ is as small as possible.) Therefore, by definition of $k_{e,i,s}$, the class $[x_{e,i}]_{E_e}$ has size $k_{e,i} = \lim_s k_{e,i,s} \geq m_e$, and by induction on e and s we know that $m_e > m_d$ for all $d < e$. On the other hand, by our choice of i, $\mathcal{R}_e$ last receives attention at some stage $s > \max\, [x_{e,i}]_{E_e}$, and so, for every $d > e$, $m_d \geq m_{d,s+1} \geq 2 + k_{e,i}$. We now show that the E-classes have size exactly $(m_d - 1)$ for $d \in \omega$. Thus E contains no class having the same size as $[x_{e,i}]_{E_e}$, and $\mathcal{R}_e$ is satisfied.

To see this fact about the sizes of the E-classes, we induct on d. Every E-class has the form $[y_d]_E$, and once chosen, y_d is never redefined in the construction. Moreover, as remarked in the construction, at stage $s+1$, y_d lies in an E-class of size exactly $(m_{d,s+1} - 1)$. Since each sequence $\langle m_{d,s} \rangle_{s \in \omega}$ converges to a finite value m_d and no other E-classes are ever created, it is clear that these values m_d are exactly the final sizes of the equivalence classes in E. The relation $m_d < m_{d+1}$ is seen to hold for all d (by an induction on

d and s). Thus E has only finite equivalence classes, and every $\mathcal{R}_e$ holds, as claimed above. Finally, this result also shows that there are arbitrarily large finite E-classes. So, even if E_e satisfies $\mathcal{R}_e$ by virtue of having an upper bound on the size of the finite E_e-classes, we still see that E_e and E cannot be isomorphic. ■

When one allows the elements of $\mathcal{E}_\infty$ into the enumeration as well, things become more feasible. In [8, Corollary 5.2], Goncharov and Knight gave a computable enumeration of $\mathcal{E}$, the family of all computable equivalence structures (including those with finite domains), simply by enumerating all c.e. subsets of a computable equivalence structure with infinitely many infinite classes and no finite classes. There may still exist a computable classification of $\mathcal{E}$, the family of all computable equivalence structures. By Theorem 5.6, if such a classification exists, one would not be able to partition it effectively into the subfamilies $\mathcal{E}_\infty$ (already effectively classified by Goncharov and Knight) and $\mathcal{E}_\omega$. Since the isomorphism problem for $\mathcal{E}$ is Π^0_4-complete, Proposition 1.5 (or even the strong version given in Corollary 3.3) does not apply to $\mathcal{E}$, and so the question of computable classifiability of $\mathcal{E}$ does not yield to any of the methods used in this article. We regard this question as challenging.

5.2 Oracle classifications

In this section, we show that there is a $\mathbf{0}'$-computable classification, by $\mathbf{0}'$-computable indices, of the family $\mathcal{E}$ of all computable equivalence structures and other related subfamilies. First, we construct such a classification for $\mathcal{E}_0$, the family of all computable equivalence structures with no infinite classes. To accomplish this, we will set $\boldsymbol{d}$ to equal the degree $\mathbf{0}'$ and apply Theorem 2.3. Specifically, We will build a $\mathbf{0}'$-computable enumeration $\langle F_e \rangle_{e\in\omega}$ of $\mathcal{E}_0$, in such a way that the isomorphism problem $\{\langle i, j\rangle : F_i \cong F_j\}$ is Π^0_3, and thus Π^0_2 relative to our oracle, and so that the other hypotheses of Theorem 2.3 are also satisfied.

To build F_e, we consider the partial computable function φ_e, writing E_e for the binary relation $\{\langle x, y\rangle : \varphi_e(\langle x, y\rangle)\downarrow= 1\}$. With a $\mathbf{0}'$-oracle, we may enumerate any witnesses which show that φ_e fails to compute a (total) equivalence relation on ω: either a pair $\langle x, y\rangle$ for which $\varphi_e(\langle x, y\rangle)\uparrow$, or pairs that witness the failure of reflexivity, symmetry, or transitivity.

To start computing F_e below $\mathbf{0}'$, we first set $x_0 = 0$, $z_0 = 0$, and $n_0 = 1$, and ask our oracle whether there exist two distinct elements of ω (including x_0 itself) that are both E_e-equivalent to x_0. If not, then $z_0 = 0$ enters $\text{dom}(F_e)$ and forms a singleton class in F_e. If so, then we increment n_0 to 2 at stage $s = 1$ and ask whether there exist three distinct such elements. This process continues until either

- we find a witness showing that E_e is not a total equivalence relation on ω, in which case the construction ends here, and F_e is a finite equivalence relation consisting of the classes already built; or

- we find the least number n_0 for which the E_e-class of x_0 fails to contain $(n_0 + 1)$ distinct elements. If this happens at stage s, then we set $z_1 = z_0 + n_0$ adjoin the numbers $z_0, \ldots, z_0 + n_0 - 1$ to $\text{dom}(F_e)$, and make them all F_e-equivalent to z_0, so that z_0 now lies in an F_e-class of size n_0 (just as x_0 does in E_e). This F_e-class will never grow any further. (Notice that z_1 is not yet in $\text{dom}(F_e)$.)

If the second possibility holds, we now continue by finding using $\mathbf{0}'$ the least $x_1 > x_0$ such that $(\forall y < x_1)\langle x_1, y\rangle \notin E_e$. We run the same process with x_1, potentially finding a number n_1 at some stage s as in the second possibility, in which case we set $z_2 = z_1 + n_1$, and make z_1 part of an F_e-class $\{z_1, \ldots, z_2 - 1\}$ of size n_1. We continue in this manner through all x_t and z_t.

Of course, the process above (for a particular x_t and z_t) could run for infinitely many stages s, if E_e is a total equivalence relation in which x_t is the least element belonging to an infinite E_e-class. If this happens, then F_e is exactly the equivalence relation defined by the process, comprising the finitely many finite classes built before we reached x_t. The same happens if E_e turns out not to be a total equivalence relation. (Indeed, F_e could turn out to be the empty equivalence structure, with domain $\emptyset$, for instance if 0 lies in an infinite E_e-class. This is why the empty structure is included in $\mathcal{E}_0$.) On the other hand, if E_e is a total equivalence relation with no infinite classes, then this process builds $F_e \cong E_e$. In all cases, F_e is an equivalence relation on an initial segment of ω and is $\mathbf{0}'$-computable uniformly in e. Thus we have a $\mathbf{0}'$-computable enumeration of $\mathbf{0}'$-indices of $\mathcal{E}_0$ (which includes all finite equivalence relations, even the empty relation.) The domain of each F_e is $\mathbf{0}'$-computably enumerable uniformly in e, but its size is not. The family of all these domains, while uniformly c.e. in $\mathbf{0}'$, is not uniformly $\mathbf{0}'$-computable.

Recall Lemmas 5.1 and 5.2, which showed that the isomorphism problem and the index set for $\mathcal{E}_0$ are both Π^0_3-complete. Since the structures F_e are only $\mathbf{0}'$-computable, we would expect the isomorphism problem $F_i \cong F_j$ to be Π^0_3 relative to $\mathbf{0}'$, which is to say, Π^0_4. However, the construction of the structures F_e has an additional feature: for each $x \in \mathrm{dom}(F_e)$, the size of the F_e-class of x is $\mathbf{0}'$-computable. We can exploit this fact to prove the following lemma.

Lemma 5.7 *With this construction, the set $I = \{\langle i, j\rangle : F_i \cong F_j\}$ is Π^0_3.*

Proof. Fixing a $\mathbf{0}'$ oracle, we show that I is Π^0_2 relative to this oracle. Notice that, whenever z_{t+1} is defined in the construction of F_i, the $\mathbf{0}'$-oracle knows the size n_t of $[z_t]_{F_i}$, since no more elements will ever join this class. That is, the function $t \mapsto n_t$ is partial $\mathbf{0}'$-computable, with domain $\{t : z_t \in F_e\}$. We will write y_t instead of z_t for elements of the equivalence structure F_j, and use m_t for the size of $[y_t]_{F_j}$. Then $F_i \cong F_j$ iff, for all finite subsets of ω $\{u_1 < \cdots < u_k\}$ and all $r > 0$ such that the construction of F_i defines $n_{u_1} = \cdots = n_{u_k} = r$, there exist $v_1 < \cdots < v_k$ such that the construction of F_j defines $m_{v_1} = \cdots = m_{v_k} = r$, and if the converse statement (with the roles of F_i and F_j interchanged) also holds. This is Π^0_2 in the constructions of F_i and F_j, which are $\mathbf{0}'$-computable uniformly in i and j. ∎

To apply Theorem 2.3, we will expand each structure F_e to an augmented structure $\widetilde{F}_e$ in a larger language. $\widetilde{F}_e$ will still be $\mathbf{0}'$-computable uniformly in e. The expanded language has unary relation symbols $R_1, R_2, \ldots$ and unary function symbols $f_1, f_2, \ldots$, along with the binary relation E from the original language. Each $\widetilde{F}_e$ will satisfy the following axioms:

$$\begin{array}{rl}
 & E \text{ is an equivalence relation.} \\
(\forall n) & (\forall x)[R_n(x) \iff ([x]_E \text{ contains exactly } n \text{ elements})]. \\
(\forall k) & (\forall x)\ [xEf_k(x)]. \\
(\forall n)(\forall j \le n) & (\forall x)(\forall y)[(R_n(x)\ \&\ xEy) \implies f_j(x) = f_j(y)]. \\
(\forall n)(\forall j < k \le n) & (\forall x)[R_n(x) \implies (f_j(x) \ne f_k(x))]. \\
(\forall n)(\forall k > n) & (\forall x)[R_n(x) \implies f_k(x) = x].
\end{array}$$

These axioms imply that if $R_n(x)$ holds, then $f_1(x), \ldots, f_n(x)$ are precisely the distinct elements of $[x]_E$ and $f_k(x) = x$ for all $k > n$. When we expand F_e to $\widetilde{F}_e$, our $\mathbf{0}'$-oracle knows the (finite) size of the E-class of each $z \in F_e$, hence can decide the unique n for which $R_n(z)$ holds and can find all the elements of $[z]_{F_e}$. We define the functions f_k, again with a $\mathbf{0}'$-oracle, so that

$f_1(z) < f_2(z) < \cdots < f_n(z)$ are the elements of $[z]_{F_e}$, and with $z = f_k(z)$ for all $k > n$, as required. This ensures that, if $F_i \cong F_j$, then there exists an isomorphism from $\widetilde{F}_i$ onto $\widetilde{F}_j$ as well. Conversely, of course, an isomorphism from $\widetilde{F}_i$ onto $\widetilde{F}_j$ must restrict to an isomorphism from F_i onto F_j.

The point of these axioms is that now each $z \in F_e$ generates its own F_e-equivalence class (and nothing more). We claim that now all hypotheses of Theorem 2.3 are satisfied by the $\mathbf{0}'$-computable enumeration $\langle \widetilde{F}_e \rangle_{e \in \omega}$. Recall that $\widetilde{F}_i \upharpoonright s$ denotes the substructure of $\widetilde{F}_i$ generated by the subset $\{0, \ldots, s-1\} \cap \mathrm{dom}(\widetilde{F}_i)$. Each $\widetilde{F}_i \upharpoonright s$ is an element of our enumeration (up to isomorphism, which is all that is necessary; it would be harmless to expand the enumeration to include not just every $\widetilde{F}_i$ but also every $\widetilde{F}_i \upharpoonright s$). Moreover, given any i and s, every $\widetilde{F}_k \upharpoonright s$ with $k < s$ is a finite equivalence structure (in our expanded language), and so, to find the j required by Assumption (1b) of the theorem, we simply form a new $\widetilde{F}$ by adjoining to $\widetilde{F}_i \upharpoonright s$ one new equivalence class of size larger than any class in every one of the $\widetilde{F}_k \upharpoonright s$. Defining the R_n and f_n on this $\widetilde{F}$ is easy, and our enumeration must include some $\widetilde{F}_j \cong \widetilde{F}$, so we pick this j along with t large enough that $\widetilde{F}_j \upharpoonright t = \widetilde{F}_j$. Clearly this j and t satisfy (1b).

Now suppose that $\widetilde{F}_i$ and $\widetilde{F}_j$ have the property that every $\widetilde{F}_i \upharpoonright s$ embeds into $\widetilde{F}_j$. This simply means that, for every $n \in \omega$, $\widetilde{F}_j$ has at least as many F_j-classes of size exactly n as $\widetilde{F}_i$ has. If the same holds with i and j reversed, then clearly $F_i \cong F_j$, and we saw above that this implies $\widetilde{F}_i \cong \widetilde{F}_j$. Thus Assumption (2a) of Theorem 2.3 holds. Also, for any s and t, we can determine from $\langle i, t, j, s \rangle$ (and our $\mathbf{0}'$-oracle) the exact number of classes of each size n in each of $\widetilde{F}_i \upharpoonright t$ and $\widetilde{F}_j \upharpoonright s$ (as well as an upper bound on the sizes we need to consider, since for each $z \in \mathrm{dom}(\widetilde{F}_i) \cap \{0, \ldots, t-1\}$ we can find the unique n for which $R_n(z)$ holds, and likewise for $\widetilde{F}_j$). From this information, it is immediate to see whether $\widetilde{F}_i \upharpoonright t$ embeds into $\widetilde{F}_j \upharpoonright s$ (since this just means that the latter has at least as many classes of each single size as the former), and also whether they are isomorphic (which is equivalent to each one embedding into the other).

Now Theorem 2.3 yields a $\mathbf{0}'$-computable classification of the (augmented) structures in the enumeration $\langle \widetilde{F}_e \rangle_{e \in \omega}$. This classification is easily stripped back down to simple equivalence structures once again, without introducing any new isomorphisms (since $F_i \cong F_j$ iff $\widetilde{F}_i \cong \widetilde{F}_j$). Thus $\mathcal{E}_0$ has a $\mathbf{0}'$-computable classification, by $\mathbf{0}'$-computable indices.

We state this result and use it to show that such classifications also exist for $\mathcal{E}$ and other related subfamilies.

Theorem 5.8 *There exists a* $\mathbf{0}'$*-computable classification, by* $\mathbf{0}'$*-computable indices, of the family* $\mathcal{E}_0$ *of all computable equivalence structures with no infinite equivalence classes. Moreover, there also exist such classifications of the families* $\mathcal{E}_n$*,* $\mathcal{E}_{\leq n}$ *(for every* $n \in \omega$*),* $\mathcal{E}_\omega$ *(the family of all computable equivalence structures with only finitely many infinite classes), and* $\mathcal{E}$ *itself, the family of all computable equivalence structures.*

Proof. For $\mathcal{E}_0$ this was established above. From the classification for $\mathcal{E}_0$, one immediately can build $\mathbf{0}'$-computable classifications of each $\mathcal{E}_n$, uniformly in n, just by adding n-many infinite classes to each member of the classification of $\mathcal{E}_0$. By the uniformity, one gets the classifications of $\mathcal{E}_\omega$ and $\mathcal{E}_{\leq n}$ for every $n \in \omega$. The $\mathbf{0}'$-computable classification of $\mathcal{E}$ itself is obtained by combining the classification of $\mathcal{E}_\omega$ with the computable classification of its complement $\mathcal{E}_\infty$ given by Theorem 1.3. ∎

References

[1] C. Ash & J.F. Knight; *Computable structures and the hyperarithmetical hierarchy* (Amsterdam: North-Holland Publishing Company, 2000).

[2] W. Calvert, V. Harizanov, J.F. Knight, & S. Miller; Index sets for computable structures, *Algebra and Logic* **45** (2006), 306–325.

[3] W. Calvert & J.F. Knight; Classification from a computable viewpoint, *Bulletin of Symbolic Logic* **12** (2006) 2, 191–218.

[4] R.J. Coles, R.G. Downey & T.A. Slaman; Every set has a least jump enumeration, *Journal of the London Mathematical Society* **62** (2000) 2 641-649.

[5] Yu.L. Ershov; Theorie der Numerierungen, *Zeits. Math. Logik Grund. Math.* **23** (1977), 289–371.

[6] R.M. Friedberg; Three theorems on recursive enumeration. I. Decomposition. II. Maximal set. III. Enumeration without duplication, *Journal of Symbolic Logic* **23** (1958) 3, 309–316.

[7] A. Frolov, I. Kalimullin, & R. Miller; Spectra of algebraic fields and subfields, *Mathematical Theory and Computational Practice: Fifth Conference on Computability in Europe, CiE 2009*, eds. K. Ambos-Spies, B. Löwe, & W. Merkle, *Lecture Notes in Computer Science* **5635** (Berlin: Springer-Verlag, 2009), 232–241. (For Appendix A, see `qcpages.qc.cuny.edu/~rmiller/research.html`.)

[8] S.S. Goncharov & J.F. Knight; Computable structure and non-structure theorems, *Algebra and Logic* **41** (2002) 6, 351–373.

[9] S.S. Goncharov, S. Lempp, & D.R. Solomon; Friedberg numberings of families of of n-computably enumerable sets, *Algebra and Logic* **41** (2002) 2, 81–86.

[10] D.R. Hirschfeldt, B. Khoussainov, R.A. Shore, & A.M. Slinko; Degree spectra and computable dimensions in algebraic structures, *Annals of Pure and Applied Logic* **115** (2002), 71–113.

[11] A. Kach, K. Lange, & D. Solomon; Degrees of orders on torsion-free abelian groups, *Annals of Pure and Applied Logic* **164** (2013) 7-8, 822–836.

[12] L. Kronecker; Grundzüge einer arithmetischen Theorie der algebraischen Größen, *J. f. Math.* **92** (1882), 1–122.

[13] R. Miller; $\boldsymbol{d}$-Computable categoricity for algebraic fields, *The Journal of Symbolic Logic* **74** (2009) 4, 1325–1351.

[14] R. Miller & A. Shlapentokh; Computable categoricity for algebraic fields with splitting algorithms, to appear in *Transactions of the American Mathematical Society.*

[15] M. Rabin; Computable algebra, general theory, and theory of computable fields, *Transactions of the American Mathematical Society* **95** (1960), 341–360.

[16] R.I. Soare; *Recursively Enumerable Sets and Degrees* (New York: Springer-Verlag, 1987).

[17] R.M. Steiner; Effective algebraicity, *Archive for Mathematical Logic* **52** (2013), 91–112.

Department of Mathematics
Wellesley College
106 Central Street
Wellesley, Massachusetts 02481 U.S.A.
E-mail: klange2@wellesley.edu

Department of Mathematics
Queens College – C.U.N.Y.
65-30 Kissena Blvd.
Flushing, New York 11367 U.S.A.
Ph.D. Programs in Mathematics & Computer Science
C.U.N.Y. Graduate Center
365 Fifth Avenue
New York, New York 10016 U.S.A.
E-mail: Russell.Miller@qc.cuny.edu

Department of Mathematics
Vanderbilt University
1326 Stevenson Center
Nashville, Tennessee 37240 U.S.A.
E-mail: rebecca.m.steiner@vanderbilt.edu