ENUMERATING ABELIAN p-GROUPS

ROD DOWNEY, ALEXANDER MELNIKOV, AND KENG MENG NG

ABSTRACT. Given an integer $n > 0$ we give a computable injective listing of the isomorphism types of all computable abelian p-groups of Ulm type $\leq n$. We prove a similar result for certain classes of profinite groups.

1. INTRODUCTION

A central problem in many areas of mathematics is the *classification* problem. For a class of structures $\mathcal{K}$, this problem typically asks "Is there a way to understand or classify the structures in $\mathcal{K}$ up to isomorphism?" Often this classification involves determining *invariants* which transform the question of whether $A \cong B$ into whether A and B have the same invariants. Since "isomorphism type" is itself an invariant, we would expect useful invariants to make the classification problem simpler. Examples of classes with useful invariants include dimension for vector spaces, Baer invariants for completely decomposable groups [Bae37], and the Jones polynomials for knots. Mathematical logic provides the tools for understanding the isomorphism problems for various classes of structures. As we describe below, we can use mathematical logic to clarify what we mean when we say "invariants should make the classification problem simpler," and also to formalise what it means to have no simplifying invariants for a given class. Where classification is possible, we can use logic to calibrate precisely how hard the isomorphism problem is. In this article we use tools of computable structure theory [AK00, EG00] to produce a fine-grained *algorithmic* classification for a broad class of groups.

Computable structure theory offers several general approaches to the classification problem for a given class of structures; see [GK02]. Here we recall that all typical countable structures met in practice are naturally computably given, where a countable algebraic structure is computable if its domain and the operations are Turing computable [Mal61, Rab60]. The standard representation of the additive group of the rationals is computable, and a finitely presented group is computable if its word problem is decidable. The nicest classification of an isomorphism problem is one where we can decide if two structures are isomorphic; the isomorphism problem is algorithmically decidable. For instance, the isomorphism problem for finite abelian groups is clearly decidable, and furthermore all isomorphism types of such groups can be computably listed without repetition. Although the isomorphism problem for arbitrary finite groups is also decidable, it is still open whether it is computationally *feasibly* decidable; meaning that we can decide isomorphism in polynomial time, see, e.g., [AT11].

The first two authors were partially supported by the Marsden Fund of New Zealand. The third author is partially supported by MOE2015-T2-2-055 and RG131/17.

For many infinite structures we have no hope of deciding the isomorphism problem, let alone feasibly deciding it. In these cases we seek to understand how hard the isomorphism problem is using various hierarchies or by asking whether there is a way to injectively list the isomorphism types. A somewhat counter-intuitive fact is that a class can have a computable listing of isomorphism types even though the isomorphism problem in the class is not decidable; an elementary example is the class of countable vector spaces over a fixed computable field. After understanding the problem for computable structures, the process of *relativisation* to an arbitrary oracle allows us to understand the general isomorphism problem for arbitrary countable structures. We now elaborate on these concepts in more detail.

The *isomorphism problem* for a class $\mathcal{K}$ of structures is the set $\{\langle i, j\rangle \mid \mathcal{A}_i \cong \mathcal{A}_j\}$ for some enumeration $\{\mathcal{A}_e \mid e \in \omega\}$ of the computable members of the class we are interested in. The complexity of the isomorphism problem for a class $\mathcal{K}$ can be measured using various hierarchies such as the arithmetical and the analytical hierarchy [Rog87]. For example, if we consider computable copies of the linear ordering of order type ω, the natural numbers, then the isomorphism problem classified by the halting problem, since access to the halting problem allows us to decide if x is the successor of y. Typically such results can be relativised to any oracle, which means that the oracle can be used as a parameter in the result. More formally, the "boldface" [Sac90] versions of such results are not restricted to computable members of the class but can cover the whole class. For example, in the case of torsion-free abelian groups the isomorphism problem is Σ^1_1-complete [DM08]. The result can be relativised to any oracle, showing that the collection of reals that naturally code torsion-free abelian groups is analytic complete [DM08]. Σ^1_1 sets are those which can be expressed as $x \in A \iff \exists f R(f, x)$ where R is some computable arithmetical relation, where the existential quantifier is quantifying over all the continuum many *functions*. Isomorphism is naturally Σ^1_1 since we are asking "is there a function obeying certain properties from W_e to W_j?" Showing the index set is Σ^1_1-complete amounts to showing that listing the isomorphism types of the structure is as hard as listing the isomorphism types of *any* countable structure. Hence, there can be *no* invariants which would simplify the isomorphism problem. This gives a computational "proof" that there are no reasonable invariants. The point is that if there are nice invariants like dimension for vector spaces, these invariants must simplify the isomorphism problem of the class. In the case of vector spaces, the problem becomes *arithmetical* since we only need two alternations of *number* quantifiers to decide what the dimension is. Hence logic blended with group theory allows us to answer Fuchs' question of whether there are invariants for general torsion-free abelian groups. In contrast, for computable completely decomposable groups [Bae37] the isomorphism problem is merely Σ^0_7 [DM14]. The result means that the Baer invariants [Bae37] for such groups are somewhat close to being Turing computable. Since Σ^0_7-hardness is an open question, we still have to understand how close to being computable they exactly are. The relativised version of this result can be stated in purely topological terms using the Borel hierarchy, confirming that the class of such groups is somewhat algebraically tame.

A nice algebraic classification sometimes leads to an injective enumeration of the isomorphism types in the class. That is, we also get a computable enumeration of computable members of the class in which isomorphism types are not repeated, and which every computable member of the class is represented. For example, we can easily enumerate all isomorphism types of finitely generated abelian groups without repetition. All such groups naturally have a solvable word problem, therefore looking at the computable members of the class is not really a restriction. The classification of finite simple groups also leads to an injective enumeration of the isomorphism types of the class. For more examples, see [LMS18]. As another example, if we consider the Downey and Melnikov [DM14] result on completely decomposable groups discussed above, with the help of a few iterates of the Halting problem, we can produce a list of all isomorphism types of computable completely decomposable groups without repetition (recall that the isomorphism problem for this class is merely Σ^0_7). In contrast, it follows from the Σ^1_1 completeness result [DM08] that no finite – indeed, no recursive transfinite – iterate of the Halting Problem is capable of enumerating all isomorphism types of computable torsion-free abelian groups without repetition.

Based on a similar intuition, Goncharov and Knight [GK02] suggested that a class of computable structures is *tame* if we have a way of *computably listing* the isomorphism types without repetition.

Definition 1.1. [GK02] We say that a class $\mathcal{K}$ of structures has a *Friedberg enumeration* if there is a computable listing $A_1, A_2, \dots$ of all isomorphism types of the computable members of $\mathcal{K}$ *without repetition.*

Formally, there is a computable listing $A_1, A_2, \dots$ of computable structures such that (i) A_i is a computable member of $\mathcal{K}$ for all i, (ii) for every $i \neq j$, we have $A_i \not\cong A_j$, and (iii) for every $B \in \mathcal{K}$, if B is computable then $B \cong A_i$ for some i.

If a class has a Friedberg enumeration, then we can regard it as "classified" in this well-defined sense, in spite of the actual isomorphism problem being possibly quite complex. The inspiration for the name "Friedberg enumeration" comes from Friedberg's proof [Fri58] that the computably enumerable sets can be listed without repetition. For sets, "isomorphism" means equality, and Friedberg's proof shows that a class of structures can have an undecidable isomorphism problem (the set $\{\langle i,j\rangle \mid W_i = W_j\}$ is Π^0_2 complete), yet there is a way to give a computable list of all its members without repetition. Another example, albeit a trivial one, is ω as a linear ordering, whose Friedberg enumeration consists of a single element, $\mathbb{N}$, yet the isomorphism problem is Σ^0_1-complete.

Goncharov and Knight [GK02] pointed out that classification via enumerations tends to be technically rather challenging. There is no algebraic structure on computably enumerable sets, yet Friedberg's original proof [Fri58] involves several techniques which were novel – if not revolutionary – at that time. Unsurprisingly, even adding very little algebraic content into the class of structures can make the Friedberg enumeration problem a lot harder.

In our previous work [DMN17] we solved a problem of Goncharov and Knight [GK02]. We produced a Friedberg enumeration of the class of computable equivalence structures. The [DMN17] result is significantly more complicated than enumerating the computably enumerable sets since determining whether two computable equivalence structures are isomorphic is Π^0_4-complete, compared to Π^0_2 for determining equality of computably enumerable sets.

In constructing a Friedberg enumeration one usually has to *dynamically measure* whether a given computable structure is isomorphic to another one. The more complex the isomorphism problem, the more difficult it is to measure this. Goncharov and Knight [GK02] suggested that there is no Friedberg enumeration of computable equivalence structures because the isomorphism problem is Π^0_4 which is very complicated to guess dynamically. Our solution [DMN17] requires essential use of an advanced priority argument, akin to the $0'''$ technique. In this paper we advance our techniques even further. Using three separate constructions combined into a single proof, we will produce Friedberg enumerations for broad classes of abelian groups in which the isomorphism problem could be at an arbitrarily high finite level of the arithmetical hierarchy.

Because of the relationship between abelian groups and equivalence structures, the Friedberg enumeration of computable equivalence structures also gives a Friedberg enumeration of all computable abelian p-groups of Ulm type 1. More specifically, given a computable equivalence relation E we can uniformly computably produce an abelian p-group A_E of Ulm type 1 as follows. For each equivalence class in E having size $\lambda \in \omega \cup \{\infty\}$ produce a cyclic or quasi-cyclic group $\mathbb{Z}_{p^\lambda}$, and then take the direct sum of all such subgroups, one for each equivalence class. It is not difficult to see that the map $E \to A_E$ is bijective on computable isomorphism types of equivalence relations and abelian p-groups of Ulm type 1. Thus, the main result of [DMN17] leads to a Friedberg enumeration of computable abelian p-groups of Ulm type 1. It is natural to ask whether computable abelian p-groups of higher Ulm type also possess a Friedberg enumeration.

The goal of the present paper is to prove the following result.

Main Theorem. *For each natural number $n > 0$ there is a Friedberg enumeration of all computable abelian p-groups of Ulm type $\leq n$.*

We remark that this gives the first known examples of algebraically nontrivial classes with infinite members having a Friedberg enumeration. We emphasize that the groups from the Main Theorem are not necessarily reduced. It can be shown that, for each fixed n, the reduced members of the class do not possess a Friedberg enumeration [GK02]. However, the fact that the groups are not reduced also adds a great deal of complexity to the argument. It is not hard to show that the isomorphism problem for groups in the theorem is Π^0_{2n+2}-complete. We use a modification of the jump inversion technique from [AKO] to partially reduce the situation to the case of equivalence structures. Our proof relies on two priority arguments which will be non-trivially combined. The main ideas in this argument are built upon our previous work on equivalence structures, as well as the work of Ash, Knight and Oates [AKO], along with some new devices we introduce here. We leave open the question of whether there is a Friedberg enumeration of the class of abelian p-groups of greater Ulm types, e.g., of all groups of type $< \omega$. Countable abelian p-groups are exactly the Pontryagin duals of abelian pro-p groups. Pontryagin duality is injective on the isomorphism types of countable abelian and compact abelian groups [Pon34]. Thus, the Ulm invariants of an abelian p-group give rise to pro-Ulm invariants of the respective pro-p dual; see, e.g., [Kie13] for an explicit definition. In [Mel17] the second author showed that the functor uniformly maps computable abelian p-groups into recursive pro-p groups [Mel17], bijectively on isomorphism types. We have:

Corollary 1.2. *For each natural number $n > 0$ there exists a Friedberg enumeration of recursive pro-p abelian groups of pro-Ulm type $\leq n$.*

1.1. **Overall structure of the paper.** The rest of the paper is devoted to our technical proof of the Main Theorem. First, in Section 2 we state and discuss all known results from the literature on computable abelian groups which will be needed in the proof. Next, in Section 3, we prove the main technical proposition from the unpublished paper [AKO]; see Proposition 2.6. Our proof of Proposition 2.6 is slightly different from the proof in [AKO]. In Section 4, we prove a certain modified version of Proposition 2.6 which will be used in our proof of the Main Theorem.

In Section 5 we informally outline the construction which produces the desired Friedberg enumeration. The construction is split into several modules and phases. In Sections 6, 7, 8, and 9 we describe and verify various submodules and parts of the construction. Finally, in Section 10, we will put all these pieces together into a formal construction and its verification.

2. Preliminaries

In this paper all groups are at most countable abelian p-groups for some fixed p. Recall that the p-height $h_p(a)$ of an element a of an abelian p-group is the supremum over all n such that $p^n y = a$ has a solution in the group. Given an abelian p-group A, define $A' = \{a \in A \,|\, h_p(a) = \infty\}$, $A^{(\delta+1)} = \big(A^{(\delta)}\big)'$, and take the intersection of A_β over $\beta < \alpha$ for a limit ordinal α. (Here A' should not be confused with the halting problem relative to A.) For a countable A, the sequence must stabilize at some countable ordinal α called the Ulm type of A; in this case $A^{(\alpha)}$ is equal to the maximal divisible subgroup of A. It is well-known that the maximal divisible subgroup of A detaches as a direct summand of A, and also itself splits into a direct sum of quasi-cyclic groups $\mathbb{Z}_{p^\infty}$. Here $\mathbb{Z}_{p^\infty}$ is the direct limit of the linearly ordered system of all finite cyclic p-groups under the natural inclusion.

2.1. **Equivalence structures and p-groups of Ulm type 1.** If $\alpha \leq 1$, meaning that $A' = A''$, then Kulikov's Criterion (see page 171 of [Kur60]) implies that the p-group A splits into the direct sum of its finite cyclic and infinite quasi-cyclic subgroups. Thus, each group G of Ulm type ≤ 1 is naturally associated with an equivalence structure E_G, as follows. The correspondence is formed by replacing a cyclic or quasi-cyclic summand $\mathbb{Z}_{p^\lambda}$ by an equivalence class of size λ. Note that this functor is well-behaved on isomorphism types because any two complete decompositions of Ulm type 1 abelian p-groups are isomorphic (as decompositions). The functor is also clearly bijective on isomorphism types.

The *Ulm factors* $A^{(\delta+1)}/A^{(\delta)}$ of A are themselves of Ulm type 1, and therefore they can be described by invariants similar to those for equivalence structures. The ordinal sequence of such invariants indexed by $\delta < \alpha$ gives the Ulm invariant of A; the invariant completely describes the isomorphism type of A [Kap69].

It is not hard to see that the functor $G \to E_G$ defined above is also bijective on *computable* isomorphism types; see, e.g., [Mel14]. It is clear that passing from an equivalence structure E to the corresponding group G_E is a uniformly effective process. In particular, it follows that the Friedberg enumeration of all computable equivalence structures produced in [DMN17] can be uniformly transformed into a Friedberg enumeration of *all* computable abelian p-groups of Ulm type 1. In this enumeration, each group has a computable complete direct decomposition into cyclic and quasi-cyclic summands.

It is far less clear that going from Ulm type 1 group G to the respective equivalence structure E_G is also *uniformly* effective; this is a relatively new result [MN18] and its proof is not entirely straightforward. Although will could avoid using the proposition below in its full power, it will be very convenient in the construction.

Proposition 2.1 ([MN18], Prop. 3.4)**.** *The functor $G \to E_G$ defined above is uniformly effective. Furthermore, regardless of the Ulm type of the input abelian p-group G, we can guarantee that the output of the uniform procedure is always an equivalence structure.*

Note that in the proposition above G does not have to be reduced or infinite. If G is not reduced or not of Ulm type 1, then E_G will have infinite classes.

Remark 2.2. *We will often use the uniformity of the correspondence $G \leftrightarrow E_G$ without explicit reference. Thus, groups of Ulm type 1 will be identified with equivalence structures (or vice versa) when it is convenient.*

2.2. **Basic trees.** While groups of Ulm type 1 are very similar to equivalence structures, groups of higher Ulm type resemble trees. We will use the technique of p-basic trees to work with abelian p-groups having Ulm type larger than 1.

Definition 2.3 ([Rog77])**.** *A p-*basic tree *is a set X together with a binary operation $p^n \cdot x$ of the sort $\{p^n : 0 < n < \omega\} \times X \to X$ such that:*

(1) *there is a unique element $0 \in X$ for which $p \cdot 0 = 0$,*
(2) *$p^k \cdot (p^m \cdot g) = p^{k+m} \cdot g$, for all $g \in X$ and $k, m \in \omega$, and*
(3) *for every element $x \in X$, there is a natural number n with $p^n \cdot x = 0$.*

If a prime p is fixed, then we think of a p-basic tree as a rooted tree with 0 being the root. Given a p-basic tree X, one obtains a p-group $G(X)$ as follows: The set $X \setminus \{0\}$ is treated as the set of generators for $G(X)$, and we add $px = y$ into the collection of relations if $p \cdot x = y$ in X. Every countable abelian p-group is generated by some p-basic tree [Rog77]. Each element of the group $G(X)$ can be uniquely expressed as $\sum_{x \in X} m_x x$, where $m_x \in \{0, 1, \dots, p-1\}$. Although we will usually deal with combinatorial trees which are subsets of $\omega^{<\omega}$, each such tree can be interpreted as a p-basic tree. Note that the root must always be in the tree, for every group must contain at least the neutral element 0.

Non-isomorphic trees can produce isomorphic p-groups. Here we will not give a complete description of the congruence relation $\sim$ on rooted trees which is defined by the rule: $T_0 \sim T_1$ if and only if the groups $G(T_0)$ and $G(T_1)$ are isomorphic. See [Rog77] for a detailed analysis of $\sim$.

Suppose that T is a p-basic tree viewed as a rooted tree. We call a finite chain of nodes *simple* if it is isolated, i.e., every node along the chain has at most one successor. Consider the following procedure:

"Take a simple chain extending $v \in T$, detach it, and
attach this chain to the root of T."

The procedure is called *stripping*. If the tree rank of v does not change after stripping, then the stripped tree T_1 and the original tree T give rise to isomorphic p-groups: $G(T_1) \cong G(T)$. This process can be iterated. Informally speaking, we can replace infinitely many simple chains at once (while preserving tree ranks), and obtain a *fully stripped tree* representing the same group. (The only restriction is that the tree-ranks of nodes in the tree must be preserved under this transformation.)

For example, a fully stripped tree for a reduced p-group of Ulm type 1 is just a collection of finite simple chains attached to 0.

Using this technique, Ash, Knight, and Oates [AKO] proved the following important result. Recall that a total function f is called *X-limitwise monotonic* if, for some total X-computable g we have $f(x) = \sup_z g(x,z)$, for all x; see [KKM13, DKT11].

Theorem 2.4 (Ash, Knight, and Oates [AKO]; Khisamiev [Khi92] and [His81] for $N = 1$)**.** *Suppose that A is a countable reduced p-group of Ulm type $N < \omega$. Then the following conditions are equivalent:*

(1) *A has a computable copy.*
(2) *A has a computable p-basic tree representing it.*
(3) (a) *For every $i < N$, the character $\chi(A_i)$ is a Σ^0_{2i+2} set, and*
    (b) *for every $i < N$, the set*

$$\#A_i := \{n : (n,1) \in \chi(A_i)\}$$

is $\mathbf{0}^{(2i)}$-limitwise monotonic.

The basis of induction in the proof of this theorem essentially says that abelian p-groups of Ulm type 1 have the same computability-theoretic and algebraic invariants as computable equivalence structures; this case is rather simple and has been known for several decades [His81], see also the surveys [Mel14, Khi98]. The case of Ulm type $n > 1$ is significantly more difficult. To explain what happens in this case, we need several definitions.

Definition 2.5. We say an abelian p-group H of Ulm type 1 *proper* if it is reduced (i.e. $H' = 0$) and furthermore the sizes of the finite cyclic summands in its full direct decomposition are unbounded in size.

The inductive step in the proof of (3) $\to$ (2) of Theorem 2.4 uses a functor that allows us to uniformly pass from a Δ^0_3 abelian group F represented by a p-basic tree and a proper H to a computable abelian p-group $T_H(F)$ with the properties $(T_H(F))' \cong F$ and $T_H(F)/(T_H(F))' \cong H$. By induction, we will have already proven that F can be represented by a Δ^0_3 p-basic tree. It is well-known that each Δ^0_3-tree can be represented as a Π^0_2-subtree of $\omega^{<\omega}$, and uniformly so. We identify F with the respective Π^0_2-tree. We also identify a proper abelian p-group H with the corresponding equivalence structure. Under this correspondence, an equivalence class of size n will represent a cyclic summand of order p^n. Recall that this correspondence is also uniformly effective; see Proposition 2.1. Having in mind the uniform correspondence $H \leftrightarrow E_H$, we will abuse notation and write H for E_H. Since H is proper, the respective equivalence structure E_H will have only finite classes, but the sizes of these classes will be unbounded.

Proposition 2.6 (Ash, Knight and Oates [AKO])**.** *There is a uniform procedure which given a computable copy of a proper H and a Π^0_2 p-basic tree F, outputs a computable p-basic tree $T_H(F)$ with the properties $(T_H(F))' = F$ and $T_H(F)/(T_H(F))' \cong H$.*

It is not difficult to see that the harder implication of Theorem 2.4 follows from the above-mentioned basic case and Proposition 2.6. Unfortunately, all known proofs of the proposition are combinatorially rather involved.

The paper [AKO] was not published because the authors learned of a similar result of Khisamiev. The problem with Khisamiev's published proof [Khi92] is that it does not use p-basic trees and is extremely hard to follow, verify, or modify. Moreover, it is not even clear if it is completely correct. Although a description of $T_H(F)$ using p-basic trees can be found in [Mel14, DMN16], none of these published descriptions can be viewed as complete proofs. We will need a minor modification of the original proof of the proposition. Thus we decided to give a detailed proof of Proposition 2.6; it is contained in Section 3 below.

3. Proof of Proposition 2.6

We split the exposition into several parts, starting from a very informal idea and then adding more details later.

3.1. **An informal idea.**

We remind the reader that $F \in \Pi^0_2$ if and only if there is a computable relation R, such that $x \in F$ if and only if $\forall s \exists t R(x, s, t)$. We say that "$x$ fires" for the first time if $R(x, s, t)$ holds for some t, and more generally, x fires the for the n-th time if it has fired n-1 times and $R(x, n, t)$ is observed to hold. Thus, $x \in A$ if and only if it fires infinitely often. If $x \in A$ then we say that the Π^0_2 outcome holds, and otherwise we say that the Σ^0_2-outcome holds.

We intend to build $T_H(F)$ as follows. If a node $x \in \omega^{<\omega}$ looks in F (when represented as a Π^0_2 p-basic tree), then we make progress in driving its tree-rank to infinity. We do this by attaching more extra finite simple chains to x when the Π^0_2-predicate describing F "fires"; the sizes of the new simple chains we attach are taken from the sizes of summands in H. (Recall that we identify H with the equivalence relation, thus the complete decomposition of H is computable.) If x looks like it is not in F we stop attaching new chains to x.

The obvious difficulties. There are of course several obvious difficulties with the rough idea outlined above. First of all, the sizes of classes/summands in H are not given as a computable or a computably enumerable set, and therefore they need to be guessed and updated at later stages of the construction after they are attached to a node. Note that they can only grow in length, for otherwise the tree will not be computable. Secondly, a class of some fixed size may occur in H more than once, and this must also be taken into account. For instance, if H has exactly 5 classes of size 3 then, up to stripping, $T_H(F)$ must also have exactly 5 chains of length 3. Finally, we could have added a few extra finite chains to x but then x will never fire again. We must understand how such Σ^0_2-outcomes will effect the isomorphism type of the output and what has to be done to control these effects.

3.2. **The elementary case: Computable sizes without repetition.**

Consider the easy but illustrative case in which the sizes of classes/summands in H have no repetition and furthermore form a computable set. Under this assumption we do not have to worry about updating the lengths of chains, and it is not necessary to monitor the multiplicity of each class/summand in H.

3.2.1. *An informal discussion.* Under the above assumptions on H it is not hard to produce $T_H(F)$ by implementing the informal idea. However, even in this simple case we must be careful of the Σ^0_2 outcome $x \notin F$. What we have is a list H of acceptable finite chains. If x fires infinitely often then we will need an infinite tree of extensions below x, thus driving the tree-rank of x to infinity. This is done using sizes in H. However, if x only fires finitely often we need to make sure that we have not introduced new paths which kill the property $T_H(F)/(T_H(F))' \cong H$. The fact that x might fire finitely often means that there will be a finite part of T which needs to correspond to simple paths of lengths in H which means that they are irrelevant after stripping.

We illustrate this situation in the following example.

Example 3.1. *Suppose $px = 0$, i.e., it is an immediate successor of the root node in $\omega^{<\omega}$. Assume that the Π^0_2 predicate has "fired" on x, i.e. a further instance of the Π^0_2 event "$x \in F$" has been observed. Suppose that we have used a simple chain of length 3 on x. This chain corresponds to a class of size 3 in H. However, imagine that we will never get to add further finite simple chains to x because the predicate will never fire again on x. As the result, we will end up with a finite simple chain of length 4. In the group that we will have constructed, it will correspond to a cyclic summand of order p^4. But there may be no equivalence class of size 4 in H, and thus $T_H(F)/(T_H(F))' \ncong H$.*

This problem is quite easy to overcome. Instead of adjoining a chain of length 3 to x, attach a chain of length 2. If later x fires again, use some longer class from H, say, of size 17. Attach a chain of length 16 to x and extend that old chain of length 2 attached to x by one extra node. This way we will form a simple chain of length 3 having a longer chain next to it. At this stage, the subgroup generated by all the mentioned nodes will be isomorphic to $\mathbb{Z}_{p^{17}} \oplus \mathbb{Z}_{p^3}$ which is consistent with the sizes in H.

There is another problematic scenario which must not be overlooked; it is explained in the example below.

Example 3.2. *At stage s we have adjoined a very long auxiliary chain to σ because the predicate has fired on σ. Suppose τ extends σ, and that τ was thought to be in F for a few stages before s. Thus, it is possible that the previous longest auxiliary chain ξ' that we saw in the construction prior to stage s was attached to τ. It must be of length $n - lth(\tau)$ corresponding to some size n in H (where $lth(\tau)$ denotes the length of τ), for otherwise we would face the problem outlined in Example 3.1 above.*

But as the result of our action on σ, up to stripping, the longest auxiliary chain attached to τ will be transformed into a simple chain of length $n - (lth(\sigma) - lth(\tau))$ and not n, thus potentially upsetting the isomorphism type of the group spanned by T (if τ never fires again). The simplest solution here would be to extend the auxiliary chain attached to τ to a slightly longer chain. This is done by lengthening it using by $lth(\sigma) - lth(\tau)$ extra nodes.

Finally, there is another situation similar to that explained in the example above which may also result in upsetting the isomorphism type of the group. Consider our final example below.

Example 3.3. *In the notation of the previous example, suppose τ is an initial segment of σ of length d. At stage s we adjoin a very long auxiliary chain to σ because the predicate fires on σ, but the previous longest auxiliary chain ξ' is attached to τ. It must be of length $n - lth(\tau)$ for some corresponding n that occurs in H.*

Up to stripping, the longest auxiliary chain attached to τ is now a simple chain of size $n - lth(\sigma)$ and not n. To fix this issue extend this auxiliary chain by $lth(\sigma)$ extra nodes, as before.

Assuming the sizes of classes in H form a computable set and have no repetition, we generalise the examples above into the simple construction below.

3.2.2. *Formal details.* Fix a computable copy of $\omega^{<\omega}$ viewed as an infinitely branching tree with its root the empty string e located at its top. Identify F with a Π^0_2-subtree of $\omega^{<\omega}$ such that each $\sigma \in \omega^{<\omega}$ has infinitely many successors which do not belong to F; furthermore, we can assume that this set of successors of σ outside F has an infinite computable subset of nodes. This subset will be used to attach external chains whose sizes will be taken from H. These external chains will be called *auxiliary*. Each auxiliary chain will be associated with exactly one class/summand in $E_H \leftrightarrow H$ having size greater or equal to the length of the auxiliary chain.

We also fix a Π^0_2 predicate S and a computable predicate R such that $S = \{x : \exists^\infty z R(x,z)\}$. Whenever a new existential witness for z is found in R, we say that S "fires" on z. We identify finite strings with their computable indices, and we also assume that S fires on σ implies that S has also fired on every predecessor of σ at least once again. Without loss of generality, assume that at every stage exactly one node of $\omega^{<\omega}$ fires. Also recall that recall the empty string e belongs to F.

Construction *(the elementary case).* Initially, at stage 0, set $U = \emptyset$ and $T_0 = T_H(F)[0] = \{e\}$. At stage s, perform the following actions.

Suppose σ has fired. By our assumption, each initial segment τ of σ fired at least once again at some earlier stage. Let $U = \{u_1, \ldots, u_s\}$ be the set of sizes in H which have been declared *used* in the construction so far.

Consider the subtree $T_{s-1} = T_H(F)[s-1]$ of $\omega^{<\omega}$ listed by the end of the previous stage s-1, and let K_s be the subtree of T_{s-1} rooted in σ (which has just fired).

(1) Fix a number m (at least twice larger than any number mentioned so far) from the set of computable sizes that occur in H.
(2) Attach a chain of length $m - lth(\sigma)$ to σ.
(3) If there is an auxiliary chain of length $n_j - lth(\sigma)$ associated with a size $n_j \in U$ and attached to σ, then enlarge this simple auxiliary chain to one of length n_j.
(4) Suppose there is an auxiliary chain ξ attached to some τ extending σ which is associated with some $n_k \in U$ but whose length is not equal n_k. If there are no such chains then do nothing. If $lth(\xi) = n_k$-$lth(\sigma)$ or longer, then again do nothing. Otherwise, suppose $lth(\xi) = n_k$-d, where $d < lth(\sigma)$ (cf. Example 3.3). In this case extend this auxiliary chain by adjoining $lth(\sigma)$-d extra consequent nodes to the end of it. (The reader will of course notice that (3) can be incorporated into (4) by allowing $\tau = \sigma$; we however feel that this would make the exposition a bit more cryptic.)

Let k be the smallest among the sizes that occur in H but has not yet been declared used in the construction. To complete the stage, adjoin a simple auxiliary chain of size k to the root e of T_{s-1}, associate the new auxiliary chain with k in H, and also enumerate k into U. Finally, define $T_H(F)[s]$ to be the extension of $T_H(F)[s-1]$; it will be equal to the collection of all auxiliary chains and their prefixes/predecessors that have been defined by the end of stage s. Go to the next stage.

Verification *(the elementary case).* It is clear that the tree-rank of a node in the fixed representation of $\omega^{<\omega}$ is infinite if and only if the node lies on the Π^0_2 subtree F. It is also clear that $T = \cup_s T_s$ is a computably enumerable subtree of $\omega^{<\omega}$; using the standard technique we can transform it into a computable one. Thus, it remains to verify that the abelian p-group spanned by T has the correct isomorphism type.

For convenience we will abuse notation and identify trees with the respective groups, and we will not distinguish equivalence structures from the respective p-groups of Ulm type 1.

We must verify that, up to stripping, the nodes of finite tree-rank form a copy of H. In other words, the fully stripped version of T must contain only finite chains of lengths that occur in H. Making sure that $T/T' \cong H$ was the main point of performing substages $(1)-(4)$ at stage s. We will argue that the actions performed at substages $(1)-(4)$ guarantee that the following properties hold:

(P1) The fully stripped version of the finite tree T_s is composed of simple chains having sizes/lengths that occur in H.
(P2) If a node $\sigma \in T_s$ is in F, then all finite auxiliary chains which are attached to σ in T_s, except for at most one (call it exceptional for σ at s), have sizes/lengths that occur in H.
(P3) If ξ is an exceptional auxiliary chain for σ in $(P2)$ at stage s, and $x \in F$, then there is a stage $t > s$ after which ξ is extended to a chain of length that is mentioned in H; after this stage this auxiliary chain will never be exceptional for σ (or any other τ) ever again.

The point of attaching a simple chain of length $m - lth(\sigma)$ to σ (and not of length m) was to ensure that the bad scenario explained in Example 3.1 does not occur in T_s. Recall that m was picked very large, and therefore $m - lth(\sigma)$ is much longer than any other chain that may currently be in T_{s-1}. Thus, after a complete stripping of T_s, this new auxiliary chain will remain attached to σ, and these two combined will form a simple chain of length m as desired.

Note that attaching a very long new auxiliary chain to σ may result in upsetting $(P1)$, as explained in Examples 3.2 and 3.3. Our actions at substages $(3)-(4)$ were essentially formalisations of the straightforward strategies outlined informally in Examples 3.2 and 3.3. Thus, a calculation of lengths of simple chains at each stage shows that (P1) holds at every stage.

To see why $(P2)$ holds, note that the predicate will fire infinitely many times on σ. Thus, there will be infinitely many auxiliary chains attached to σ in the limit. By induction on a stage, at every stage at most one such auxiliary chain attached to σ can be exceptional, i.e., may be unequal in length to the respective size in H. But at the stage at which the predicate fires for σ again this chain will be made equal to the respective size in H according to the instructions at substage (2). This proves $(P3)$.

Conditions $(P2)-(P3)$ show that, up to stripping, the nodes of infinite rank contribute only chains of lengths that occur in H. Recall that the empty string is assumed to be in F; it corresponds to zero of the group spanned by F (and by T). Thus, each eventually abandoned finite piece Γ of T (due to the Σ^0_2-outcome) is attached to some node of F. Each node of F will have arbitrarily long auxiliary chains adjoined to it, and therefore there is a stage t such that, after t, Γ can be fully stripped off T into a union of finite simple chains. By (P1), the sizes of these chains occur in H. Finally, our actions at the beginning of every stage ensured that no size that occurs in H is missed in T/T'. This finishes the verification in the elementary case when sizes of classes in H form a computable set.

3.3. **The general case.** The original proof in [AKO] used limitwise monotonic functions; recall that a function g is limitwise monotonic if $g(x) = \sup_z f(x, z)$ for some total computable f of two arguments. We will exploit the uniformity of the correspondence $H \leftrightarrow G_H$ (see Proposition 2.1) and will not distinguish between an equivalence structure H and the respective group G_H. This identification allows us to completely eliminate limitwise monotonic functions from the construction. The proof of Proposition 2.1 is not straightforward; so the combinatorics related to limitwise monotonicity has not mysteriously vanished, they just got absorbed into this proof.

3.3.1. *The main difficulty.* The difference with the elementary case of a computable set is obvious. Now, when we attach an auxiliary chain, we cannot guarantee that the size of the respective class in H is final. In particular, we may have introduced a chain ξ which was very long, but at some later stage some earlier auxiliary ξ' may outgrow ξ. This introduction of ξ results in difficulties in the spirit of Examples 3.2 and 3.3.

3.3.2. *An informal description of the solution.* Imagine that you *knew* ahead of time that the first auxiliary chain corresponds to the smallest class in H, the second auxiliary chain that we added corresponds to the second smallest, etc. For simplicity, further assume that sizes of classes in H have no repetition. Recall that the comparison of sizes of auxiliary chains is the main driving force of the construction in the elementary case of a computable set. In the verification of the elementary case we do not even use these sizes, as long as we can guarantee that the new chain is much longer than all other chains we have seen so far.

So, assume that the final sizes of classes contain no repetition and can be computably compared, even though their final sizes are merely approximable from below. In this case we would simply run the construction of the elementary case, but we will have to update the lengths of auxiliary chains when the respective classes increase in size, as follows.

If a chain ξ has to be grown larger than the current size of some ξ', but *we know* ξ' will correspond to a larger class in the limit, we just do nothing with ξ until ξ' grows too. Since there are at most finitely many such ξ', ξ'', ..., this is only a finite delay. Under this strong assumption on comparability of sizes, the construction described for the elementary case goes through with only very minor adjustments.

Of course, in general we cannot guarantee that the final sizes of classes in H can be compared effectively. However, we do know that H contains arbitrarily large finite classes. Thus, we can implement the following re-targeting procedure.

3.3.3. *An informal description of re-targeting.* At every stage each auxiliary chain ξ is associated with some class in H whose size it is monitoring, let $t(\xi)$ denote this class. Each auxiliary chain is also given an *index* according to the stage at which it is introduced, with smaller indices corresponding to earlier stages. Write $i(\xi)$ for the index of ξ.

Re-targeting: If $t(\xi)$ has increased in size then the lengths of ξ and of ξ', such that $i(\xi') > i(\xi)$ and ξ' is not attached to e, will have to be updated.

For ξ, simply add as many extra nodes as there are new points in $t(\xi)$. For each ξ' with $i(\xi') > i(\xi)$ and which is not attached to the root e, update $t(\xi')$ and set it equal to the first found new class in H which currently is larger than $t(\xi'')$ for every ξ'' with the property $i(\xi'') < i(\xi')$ (these includes ξ). Since H contains arbitrarily large classes, we keep enumerating H until such a class is found.

Since some classes may be left out of the range of t, we introduce new auxiliary chains, associate them with the missed classes, and attach them to the root e. Since the root is guaranteed to be in the tree F there is no need to be careful with the way they are approximated. In particular, we will not have to re-target these chains ever again in the future. In the general case the sizes of classes in H may of course contain repetition, but is not too problematic; in fact, we do not even have to do anything special to control the repetition. All we need to do is to make sure that h is bijective. (This is the main advantage of using an equivalence structure instead of a limitwise monotonic function.)

3.3.4. *Formal details.* Recall that we are given a Π^0_2 subtree F of a special copy of $\omega^{<\omega}$, and we also are given an abelian reduced p-group H of Ulm type 1 in which sizes of elementary cyclic summands are unbounded. As usual, H can be uniformly replaced with a computable equivalence structure; we identify H and this structure.

Construction *(the general case).* Initially, at stage 0, set $U = \emptyset$ and $T_0 = T_H(F)[0] = \{e\}$. At stage s, go through the four phases described below.

Phase 1: Updating ranks of nodes. Without loss of generality, at every stage exactly one node of $\omega^{<\omega}$ fires (recall $e \in F$). Suppose σ has fired. By our assumption, each initial segment τ of σ fired at least once again at some earlier stage. Let $U = \{u_1, \dots, u_{n(s)}\}$ be the set of classes in H which are currently in the range of h, and let $\xi_1, \xi_2, \dots, \xi_{n(s)}$ be simple auxiliary chains with $t(\xi_i) \in U$ and having indices $1, 2, \dots, n(s)$, respectively.

Consider the subtree $T_{s-1} = T_H(F)[s-1]$ of $\omega^{<\omega}$ enumerated at the end of the previous stage s-1, and let K_s be the subtree of T_{s-1} rooted in σ (which has just fired).

(1) Fix an m larger than any number mentioned so far and so that m is equal to the size of some class of H which is currently outside of the range of t; if no such large class is seen in H at the stage, do several extra steps in the enumeration of H until such a class is found.
(2) Attach a new chain ξ_{s+1} of length $m - lth(\sigma)$ to σ.
(3) If there is an auxiliary chain of length $n_j - lth(\sigma)$ associated with a size $n_j \in U$ and attached to σ, then enlarge this simple auxiliary chain to one of length n_j.
(4) Suppose there is an auxiliary chain ξ attached to some τ extending σ which is associated with some $n_k \in U$ but whose length is not equal n_k. If there are no such chains then do nothing. If $lth(\xi) = n_k$-$lth(\sigma)$ or longer, then again do nothing. Otherwise, suppose $lth(\xi) = n_k$-d, where $d < lth(\sigma)$ (cf. Example 3.3). In this case extend this auxiliary chain by adjoining $lth(\sigma)$-d extra consequent nodes to the end of it.

Phase 2: Re-targeting. Suppose $i < s$ is least such that $t(\xi_i)$ has grown in H since the previous stage. For ξ_i, add as many extra nodes as there are new points in

$t(\xi_i)$. For each $j > i$ and which is not attached to the root e, update $t(\xi_j)$ and set it equal to the first found new class in H whose index is larger than the index of the current $t(\xi_j)$ and which currently is larger than $t(\xi_k)$ for every $k < j$; enumerate H until such a class is found.

Phase 3: Bookkeeping. Let u be the smallest among the classes that occur in H which is currently outside of the range of h. Adjoin a simple auxiliary chain ξ of length $k = card(u)[s]$ to the root e of T_{s-1}, set $h(\xi) = u$, and also enumerate ξ into U.

Verification *(the general case).* It is again clear that the nodes which will end up having infinite rank are exactly the nodes of F, therefore T' has the correct isomorphism type. Also, T is clearly a computably enumerable subtree of $\omega^{<\omega}$; it can be easily transformed into a computable tree. We must argue that $T/T' \sim H$.

By induction on a stage and on the index of a simple chain ξ we can show that $h(\xi)$ is stable. Indeed, $h(\xi)$ has to be changed only if a chain of a smaller index has to be grown. Since all classes in H are finite and by the inductive hypothesis, there are only finitely many stages at which $h(\xi)$ has to be changed. Suppose $h(\xi)$ settled on some class u in H. Go to the stage at which the size of u reaches its final value k. After this stage we have $lth(\xi) \leq card(u) = k$, and it may be smaller due to its position in T and because of the stripping issues which we explained in detail in the elementary case. However, it cannot outgrow k and, thus, it eventually settles.

Phase 3 was responsible for making sure that no class of H is left without an auxiliary chain associated to it. Note that chains attached to the root e cannot be re-targeted again. We are guaranteed that e will have arbitrarily long finite chains attached to it, and therefore there is no need to worry about any stripping issues. We explicitly made sure that every class which could potentially be without h-preimage will eventually be permanently associated with an auxiliary chain attached to e. Combined with the inductive argument above, this implies that every class in H will eventually be permanently associated with an auxiliary chain in T, and this correspondence is 1-1.

The rest of the verification is very much similar to the elementary case when the sizes of H form computable set. We must verify that the following conditions hold:

(P1) If a node $\sigma \in T_s$ is in F, then all finite auxiliary chains which are attached to σ in T_s, except for at most one (call it exceptional for σ at s), have their lengths equal to sizes of classes that occur in H_s.

(P2) If ξ is an exceptional auxiliary chain for σ in $(P2)$ at stage s, and $x \in F$, then there is a stage $t > s$ after which ξ is extended to a chain of length that is mentioned in H; after this stage this auxiliary chain will never be exceptional for σ (or any other τ) ever again.

Condition $(P1)$ is explicitly maintained at every stage. For a given σ, there is at most one exceptional ξ whose length is lagging behind the size of $h(\xi)$ according to the instructions in Phase 1. To see why $(P2)$ holds, go to the stage at which the length of ξ reaches its final value. Since $\sigma \in F$, there is a longer chain which will eventually be attached to the same σ. Thus, according to the instructions at substage (3) of Phase 1, the length of ξ must be set equal to the size of $h(\xi)$.

It remains to consider what happens with nodes which are forever abandoned because they never fire again. Let σ be such a node, and assume its predecessor is in F. Then there are at most finitely many auxiliary chains attached to it or its successors. Go to the stage at which all of these chains reach their final value. The instructions of Phase 1 guarantee that after full stripping this segment of the tree becomes a collection of disjoint simple chains having lengths equal to sizes of the respective classes in H. Also, recall that Phase 3 guarantees that no classes are left without h-preimage. Combined with $(P1)$ and $(P2)$, this shows that $T/T' \sim H$. This finishes the proof of Proposition 2.6.

3.4. **Properties of the construction.** The following properties of the construction from the proof of Proposition 2.6 will be quite important later:

Property 3.4. Whenever a simple auxiliary chain obtains a new image in H, the chain grows in size.

Property 3.5. A chain $\mathcal{C}$ is re-targeted only if some earlier introduced chain has grown.

Property 3.6. We re-introduce (the size of a) class in H that has been abandoned due to re-targeting, as follows. We attach a new simple auxiliary chain of the correct length to the root and associate it with the class. *The new simple chain will never be re-targeted again.*

Note also that the proof above does not assume or use that F corresponds to a reduced abelian group. This implies:

Theorem 3.7. *Suppose A is an abelian p-group of Ulm type > 1 which is not necessarily a reduced group. Then the following are equivalent:*

1. *A has a computable copy;*
2. *A' has a Δ^0_3-copy and A/A' has a computable copy.*

Proof. (2) $\to$ (1). Recall A has Ulm type > 1, and therefore A/A' is infinite and furthermore the sizes of cyclic summands in A/A' are unbounded, for otherwise every element of infinite height in A would have to be divisible. We can therefore run the proof of Proposition 2.6 which does not require the p-basic tree for A' to be well-founded.

(1) $\to$ (2). This is the same as in the case when A is reduced [AKO]; the key observation here is that the proof of this implication does not need the group to be reduced provided that A' is not divisible. Since this proof has never been published and the proof in [Khi92] uses a different notation, we give our version of this proof below.

Since the Ulm type of A is at least 2, there must be an element $a \in A'$, $a \neq 0$, which is not divisible; equivalently, any p-basic tree of A' must have a non-trivial terminal node, for otherwise A' would be divisible and $A' = A''$, contradicting the assumption. This means that a has infinite p-height in A, but there is no x with the property $px = a$ which also has infinite p-height. Using a, define a limitwise monotonic function f, as follows. List all $x_1, x_2, \ldots$ with the property $px_i = a$ and define $f(i) = h_p(x_i) + 1$, where $h_p(x_i)$ stands for the p-height of x_i.

We claim that the range of f is infinite and is contained in the collection of all n such that that A/A' has a cyclic summand of order p^n. We verify this claim in the paragraph below.

It is clear that the range of f is infinite, for the p-height of a is infinite but it is not divisible. Since the heights of the x_i are unbounded, for each i there will be a j with $h_p(x_j) > h_p(x_i)$; this will imply $h_p(x_i - x_j) = h_p(x_i)$, because $h_p(x_i) = h_p(x_j +$

$(x_i - x_j)) \geq \inf\{h_p(x_j), h_p(x_i - x_j)\}$ and $h_p(x_i - x_j) \geq \inf\{h_p(x_i), h_p(x_j)\} = h_p(x_i)$. Note that $p(x_i - x_j) = 0$, and for some α we have $p^{h_p(x_i - x_j)}\alpha = (x_i - x_j)$. But this makes $\langle \alpha \rangle$ a pure cyclic subgroup of A of order $h_p(x_i - x_j) = h_p(x_i)$ (that is, for each x in the subgroup its p-height in A is witnessed within the subgroup), and pure cyclic subgroups detach [Fuc70], so $A \cong B \oplus \langle \alpha \rangle$. Since $(C \oplus D)' = (C' \oplus D')$ and $\langle \alpha \rangle' = \langle \alpha \rangle$, we have $A/A' \cong B' \oplus \langle \alpha \rangle$, thus proving the claim.

Note that, essentially, we have just showed that if A has a cyclic summand of order p^n then A/A' also has a cyclic summand of order p^n. In fact, the converse implication is also true. To see why, suppose $\langle \alpha \rangle$ of order p^n detaches in A/A'. The coset of α must contain an element a such that $p^n a$ has infinite height, but $p^m a$ has finite height for each $m < n$. Also, if the p-height of a in A was not zero then, for some $b \in A$, we would have $pb = a$ which would also hold modulo A'. So for some β we would have $p\beta = \alpha$, contradicting the choice of α. The same argument shows that the p-height of each $p^m a \in \langle a \rangle$, $m < n$, is equal to the p-height of its coset in A/A' and is equal to m. Since the p-height of $x = p^n a$ is infinite, there exists some c with the property $p^n c = x$ and with $h_p(c) > 0$. Consider the element $y = a - c$ and the cyclic subgroup $\langle y \rangle$ of A. Then $h_p(y) = 0$, for otherwise $h_p(a) = h_p(y + c) \geq \inf\{h_p(y), h_p(c)\} > 1$ would contradict $h_p(a) = 0$. Similarly, for $m < n$, $h_p(p^m y) = m$; otherwise $h_p(p^m a) = h_p(p^m y + p^m c) \geq \inf\{h_p(p^m y), h_p(p^m c)\} > m$ would contradict $h_p^A(p^m a) = h_p^{A/A'}(p^m \alpha) = m$. This shows that $\langle a \rangle$ is pure in A and thus detaches as a direct summand of A.

So cyclic direct summands are the same in A and A/A'. This makes the set

$$\#A = \{\langle m, n \rangle : A/A' \text{ has at least } m \text{ cyclic summands of order } p^n\}$$

a Σ^0_2-set. Indeed, it is sufficient to search for $\mathbb{Z}_p$-independent $\alpha_1, \ldots, \alpha_m$ of order p such that, for each $i \leq m$, $h_p(\alpha) = n$; the latter requires $0'$. With the help of $\#A$ and the limitwise monotonic f defined above, we can use the standard techniques (e.g., [KKM13]) to produce a computable presentation of the equivalence structure $E_{A/A'}$ and, thus, of A/A'. $\square$

Remark 3.8. *The theorem above fails for non-reduced groups of Ulm type 1. Indeed, it is not difficult to build a computable non-reduced abelian group of Ulm type 1 such that its reduced component has no computable copy. Equivalently (Prop. 2.1), there exists a computable equivalence relation E such that the subrelation $F(E)$ consisting of exactly the finite classes of E does not have a computable copy. It is essentially sufficient to produce a Σ^0_2 set which is not limitwise monotonic* [His81, KNS97].

Remark 3.9. *The functor witnessing the proof of* (2) $\to$ (1) *is uniform if we guarantee that A/A' has only finite summands whose orders are* not *uniformly bounded.*

The "injury" in the construction of Ash, Knight, and Oates is at most finite. Our next task is to understand what happens when H in $T_H(F)$ is not necessarily reduced; that is, when E_H contains infinite classes.

4. The modified Ash-Knight-Oates strategy

Suppose H is an equivalence structure. We identify H and the respective G_H which is a direct sum of cyclic or quasi-cyclic p-groups. Recall that H is proper if it has only finite classes, but the sizes of the classes are not uniformly bounded.

We will use a modification of the original Ash-Knight-Oates functor $T_H(F)$ which handles the case when H is not necessarily proper, but only if almost all classes of an improper H are infinite. In [DMN17] such equivalence structures were given the following disparaging name.

Definition 4.1. Call a computable equivalence structure *an infinite junk* if it has infinitely many classes almost all of which are infinite.

We abuse notation and write $T_H(F)$ for the *modified* Ash-Knight-Oates jump inversion which is described in the lemma below.

Lemma 4.2. *There is a uniform procedure which, on input a computable copy of an equivalence structure H and a p-basic tree F represented as a Π^0_2-subtree of $\omega^{<\omega}$ with $e \in F$, outputs a computable p-basic tree $T_H(F)$ with the properties:*

(1.) *If H is proper then $(T_H(F))' = F$ and $T_H(F)/(T_H(F))' \cong H$.*
(2.) *If H is an infinite junk, then $T_H(F) \cong H$.*
(3.) *If H is finite then $T_H(F)$ is finite, and furthermore its cardinality can be assumed arbitrarily large and with all possible uniformity.*

Note that there are no assumptions on F apart from $e \in F$, which is equivalent to saying that 0 is in the subgroup generated by the p-basic tree F, and therefore this assumption is satisfied without any loss of generality.

Proof. We adopt the following modification to the original strategy of Ash, Knight, and Oates:

Modification 1. *At every stage at which the Ash-Knight-Oates module initiates a new search through H or makes a change to its p-basic tree, adjoin a very long simple chain never seen so far to the root of the p-basic tree. Call this extra simple chain* ***subsidiary****. If the subsidiary chain has just been introduced, then it does not have to copy any class in H. We also initiate a search for a new and large enough class in H that can be matched with the subsidiary chain in the future. The module will not act again until the search is finished (if ever). When the module acts again (if ever) the chain is handled as a standard auxiliary chain attached to e.*

The module will be later associated with a node on the tree of strategies, and in particular it may be initialised. We also attach a very long subsidiary chain to the root of the p-basic tree previously handled by the strategy if the strategy τ gets initialised. Since the old p-basic tree will be forever abandoned by the strategy, in this case there is no need to search for an image for the subsidiary chain in H (the image can be larger than the length of the chain).

(1.) Since H is proper, we will eventually succeed in finding a long enough class in H that can be matched to the subsidiary chain; the class in H may be (currently) larger than the chain. Once this is done, the chain becomes indistinguishable from the other many simple chains that we attach to the root 0 according to the non-modified instructions. There are no further interferences of the modification with the rest of the module. It follows that in the case when H is proper, the verification of the new module is almost literally the same as the verification contained in the previous section.

(2.) Here the modification plays no significant role either. However, the analysis of this scenario is new because the case of a non-reduced H has never been considered in the literature. Recall that the first few classes of H could be finite, but the rest of the classes are infinite, and there are infinitely many of them.

First, we claim that almost all auxiliary or subsidiary simple chains that we ever attach become infinite. Note that a simple chain may never find a stable pre-image among classes in H. However, at each intermediate step we always succeed in finding a long enough class in H to match with the chain. Whenever we switch, the chain itself must grow; see Property 3.4. Thus, we still grow the length of the chain to infinity, even though it may never find a stable image in H. Now consider those simple chains which do find a stable match in H. Almost all of these chains grow infinite by simply copying the respective stable class in H. The analysis also applies to the subsidiary simple chains from the modification. In particular, since we are never stuck at any intermediate step, there are infinitely many such infinite simple chains to be attached to the root. It follows that the divisible part of $T_H(F)$ has infinite rank.

There are at most finitely many exceptional chains that correspond to the finite classes in H. There may also be several finite configurations that become simple chains after stripping the tree. The latter corresponds to parts of the tree being forever abandoned in a Σ^0_2-outcome of the Π^0_2-approximation. Every individual simple chain, as well as each chain involved into an "abandoned" configuration, must grow whenever its image in H switches (Property 3.4). Thus, a chain or a configuration of chains can be finite only if each auxiliary chain involved into the configuration finds a stable image in H. There are only finitely many finite classes in H, and thus the reduced part of $T_H(F)$ must be finite. Furthermore, we may be forced to switch the image of a given chain only due to some currently shorter class of a smaller index has grown (Property 3.5).

If a finite class in H is skipped in the construction due to re-targeting, then it will be re-introduced again in the form of a simple chain attached to the root (Property 3.6). There are only finitely many classes having a smaller index than the index of the finite class. Therefore, by induction, each finite class will eventually find a stable image in the tree, which will be a simple chain of the correct length. It follows that the reduced part of $T_H(F)$ is isomorphic to the reduced part of H (viewed as a p-group).

(3.) This is obvious from the description of the modification, because the subsidiary chain can be taken to be arbitrarily long. It is crucial that the chain does not have to copy any class in H at the stage when it is first introduced. □

5. A plan of the proof

Recall that we have to produce an effective uniform 1-1 enumeration of all computable isomorphism types of abelian p-groups of Ulm type $\leq n$; we call such enumerations Friedberg.

If $n = 1$ then there exists Friedberg enumeration of all computable equivalence structures [DMN17]. The uniformity of the correspondence $E \leftrightarrow G_E$ gives a Friedberg enumeration of all abelian p-groups of Ulm type ≤ 1. It is clear that the groups in the list are uniformly represented by computable p-basic trees which are inherited from the full decomposition induced by the corresponding equivalence structure.

Therefore, assume $n > 1$ throughout the rest of this paper. Inductively, fix a Friedberg enumeration $(F_i)_{i\in\omega}$ of all isomorphism types of $0''$-computable abelian

p-groups of Ulm type $\leq n-1$; furthermore, assume that they are represented by Π^0_2 p-basic trees whose indices $h(1), h(2), h(3), \dots$ are given uniformly.

Remark 5.1. *The function h is computable and not merely $0''$-computable. It returns the index of the computable R_i such that $\sigma \in F_i \iff \exists^\infty z R_i(\sigma, z)$. As we noted before, it is well-known that there is a uniform procedure that transforms a Δ^0_3-tree into a Π^0_2-subtree of $\omega^{<\omega}$.*

Theorem 1.2 of [DMN17] says that there is a Friedberg enumeration $(E_i)_{i\in\omega}$ of all infinite equivalence structures, and thus of infinite abelian p-groups of Ulm type 1. This is *not* an immediate corollary of the existence of a Friedberg enumeration of all equivalence structures. Although using infinite equivalence structures is not essential for our proof, it will be convenient.

Based on the Friedberg enumerations $(F_i)_{i\in\omega}$ and $(E_j)_{j\in\omega}$ described above, fix the effective listing $(F_i, E_j)_{i,j\in\omega}$.

5.1. **Proof idea.** All informal explanations contained in this section will be clarified in the later sections. The main goal of this subsection is to informally explain some key ideas behind the formal construction.

In the notation above, suppose F_i is well-founded and E_j has only finite but arbitrarily large classes; we call such F_i and E_j *true* and *proper*, respectively. Under these assumptions we can uniformly produce a computable abelian p-group $T_{E_j}(F_i)$ of Ulm type at most n such that $(T_{E_j}(F_i))' \cong F_i$ and $T_{E_j}(F_i)/(T_{E_j}(F_i))' \cong E_j$; this is Theorem 3.7 and Remark 3.9. Since $(F_i)_{i\in\omega}$ and $(E_j)_{j\in\omega}$ are Friedberg, the Ulm classification theorem implies that unequal pairs correspond to non-isomorphic groups *provided that these pairs consist of true and proper members, respectively.* Furthermore, the Ulm classification theorem and Theorem 3.7 imply that each computable group of Ulm type k, $1 < k \leq n$, has the form $T_E(F)$ for some true F of type $< n$ and proper E having the correct complexities (Π^0_2 and computable, respectively).

To succeed in producing the desired Friedberg enumeration, we merge two $0'''$ constructions – one from [DMN17] for Ulm type 1 groups and the other for higher Ulm types $\leq n$ – and let them share the "junk". The rough idea is as follows. Given (F_i, E_j), guess trueness and properness, and simultaneously attempt to enumerate $T_{E_j}(F_i)$. If *all* F_i and E_j in the list were true and proper, respectively, then $T_{E_j}(F_i), i, j \in \omega$, would be a Friedberg enumeration of all computable abelian p-groups of Ulm type $1 < k \leq n$. Merging it with the Friedberg enumeration of all computable abelian p-groups of Ulm type 1 from [DMN17] we would get the desired 1-1 list of all groups of types $\leq n$.

However, if F_i is not true or E_j is not proper, we cannot guarantee that $T_{E_j}(F_i)$ will have Ulm type > 1. This will conflict with the enumeration of all groups of type 1. Nonetheless, by carefully controlling the group produced in each of these two unpleasant outcomes it is possible to incorporate this group of Ulm type 1 into the dynamic procedure of enumerating of all type 1 groups from [DMN17]. We will of course explain the construction from [DMN17] in sufficient detail, but delay this until §6, discussing the ideas first.

5.2. **The global architecture of the proof.** We give a more detailed scheme of the construction which will hopefully help the reader to understand the complex architecture of the proof. The construction will consist of three main modules.

(1) *The main module.* On input F_i and E_j, it performs the following tasks:
   - It measures whether F_i is true and E_j is proper. The combined complexity of these two guessing procedures is Σ^0_4 (to be verified), and it will be split into infinitely many Π^0_3-instances, one for each potential $\exists$-witness z in $\Sigma^0_4 = (\exists z)\Pi^0_3(z)$.
   - It attempts to build $T_{E_j}(F_i)$. If F_i is true and E_i is proper then, for exactly one z, exactly one submodule σ associated with (i, j, z) succeeds in building $T_{E_j}(F_i)$ of Ulm type > 1. This occurs only if the Π^0_3-predicate holds, and E is "true". The submodule σ also has several outcomes which depend on the isomorphism type of E and also on how exactly the Π^0_3-predicate fails. Under these outcomes either finite groups/structures or infinite junk structures (Def. 4.1) of Ulm type 1 are produced. They are placed into the junk collector; see the third main module below.
   - To make the structures produced below the Π^0_2- and Σ^0_2-outcomes easier to handle via Lemma 4.2, the procedure associated with σ uniformly replaces E_j with a certain H_j and works with $T_{H_j}(F_i)$ instead of $T_{E_j}(F_i)$. The equivalence structure H_j has several convenient combinatorial properties (to be explained), and of course $H_j \cong E_j$ if the latter is proper, thus $T_{H_j}(F_i) \cong T_{E_j}(F_i)$ in the Π^0_3 outcome.

(2) *The module enumerating Ulm type 1 groups.* This is literally the same as the one in [DMN17], but with equivalence structures uniformly replaced by the respective Ulm type 1 abelian p-groups. Various sub-strategies are put together into a tree of strategies, in which the true path will be $0'''$-computable. The tree produces an enumeration of all equivalence structures which mentions all structures having arbitrarily large finite classes exactly once; it also enumerates *some* isomorphism types of infinite junk and finite structures. The latter two are placed into the junk collector (see (3) below) which ensures all infinite junk and finite structures are mentioned exactly once up to isomorphism. The only missing isomorphism types are:
   - Equivalence structures having finitely many classes and at least one of these is infinite.
   - Equivalence structures which have infinitely many classes and are eventually bounded; that is, almost all classes are less in size than some fixed bound k specific to the structure.

   The uniform Friedberg list of such structures can be easily produced independently and later adjoined to the Friedberg enumeration of the rest.

(3) *The junk collector.* It is responsible for enumerating all infinite junk and finite equivalence structures/groups without repetition. Its actions are global. It handles the infinite junk and finite equivalence structures/groups produced by the two main modules as described above, and it also be introduces its own ones to make sure that the enumeration is 1-1 and surjective on isomorphism types of infinite junk and finite equivalence structures/groups. The junk collector module has two submodules:
   - *The infinite junk collector.* It is responsible for making sure that all computable isomorphism types of infinite junk structures/groups are listed, and without repetition. Its unsuccessful attempts result

in abandoning a structure in the process; abandoned structures are permanently placed into the *finite junk collector*.

- *The finite junk collector.* Its task is to ensure all finite equivalence structures/abelian p-groups are mentioned in the list, and exactly once. Several simple tricks and the movable markers technique are sufficient to sort out the combinatorics. (One such trick is described in Modification 1 in Lemma 4.2.)

The construction will be described in Section 10, but we outline it below. The construction will be split into three relatively independent phases;

(1) Phase 1 is responsible for enumerating all Ulm type $k > 1$ ($k \leq n$) groups, all groups of Ulm type 1 having arbitrarily large finite cyclic summands, and *some* finite and infinite junk groups. At this phase of the construction *the main module* and *the module enumerating Ulm type 1 groups* act simultaneously and independently according to their instructions. We ensure that there is no interactions between these two modules.
(2) Phase 2 is responsible for expanding the output of Phase 1 so that the new enumeration also contains all isomorphism types of infinite junk structures. This is done using *the infinite junk collector*.
(3) Phase 3 transforms the output of Phase 2 into an enumeration which additionally mentions every isomorphism type of a finite abelian p-group exactly once. This is done using *the finite junk collector*.

Finally, to get the desired Friedberg enumeration we merge the output of Phase 3 with the Friedberg enumeration of all eventually bounded equivalence structures and all equivalence structures having finitely many classes at least one of which is infinite; the latter of course are uniformly replaced with the respective abelian p-groups. This finishes the informal outline of the construction.

One crucial observation is that, from the perspective of the junk collector, the products of Π^0_2 and Σ^0_2 outcomes of submodules of the main module (1) are not really special when compared with similar outcomes of the module (2) taken from [DMN17]. Thus, the junk collector and the tree-construction from [DMN17] can be adopted with no modification, but all equivalence structures should be uniformly replaced with the respective Ulm type 1 abelian p-groups (Proposition 2.1).

Of course, there are many details that need to be formally and carefully clarified and verified. Nonetheless, provided that each of the three main modules succeeds in its proposed task we shall end up with a Friedberg enumeration of all computable abelian p-groups of Ulm type $\leq n$.

Section 6 contains a detailed exposition of the basic strategy for main module. It relies on the modified Ash-Knight-Oates strategy and on properties of a certain transformation $E \to H$ which is verified in Section 7. The second and third main modules can be taken from [DMN17]; no further modification to these modules is necessary in our proof. Thus, our exposition of these two modules (Sections 8 and 9, respectively) is relatively compressed. The formal construction and its verification is contained in Section 10.

6. The basic strategy

6.1. **True and proper groups.** Recall that the Ulm type of each F_i is at most $n-1$, and that each F_j is a Π^0_2 subtree of $\omega^{<\omega}$ whose index is given uniformly. Each E_j is a computable *infinite* equivalence structure which can be viewed as a an abelian p-group of Ulm type 1 in which a complete decomposition is known.

We identify E_i with the corresponding abelian p-group. According to our terminology, E_i is *proper* if it consists only of finite classes and the sizes of its classes are unbounded.

Definition 6.1. Let F be a p-basic tree. If F has a non-zero terminal node then we say that F is *true*. Note that this is equivalent to saying that the reduced part of the corresponding p-group is non-trivial.

Lemma 6.2. *Let $(E_i)_{i\in\omega}$ and $(F_i)_{i\in\omega}$ be uniform enumerations of computable equivalence structures and Π^0_2 trees as defined above.*

1. *The property "E_i is proper" has complexity Π^0_3.*
2. *The property "F_i is true" has complexity Σ^0_4.*

Proof. For (1), just state that each class is finite (Π^0_3) and that there are arbitrarily large classes (Π^0_2). The statement "F_i is true" can be described by the formula:

$$(\exists x)\,[x \in F_i \ \wedge \ x \neq e \ \wedge \ (\forall y)(y \supset x \rightarrow y \notin F_i)]$$

which gives an upper bound of Σ^0_4 for (2). □

It is not difficult to show that the bounds in the lemma above are optimal, and therefore the complexity of our guessing cannot be simplified.

6.2. **Guessing trueness and properness.** Given (F_i, E_j) we need to test whether F_i is true and E_j is proper. We suppress the subscripts in F_i and E_j and write (F, E) throughout the rest of this subsection.

We start with the simpler Π^0_3 guessing properness of E. We index classes of a computable equivalence structure by natural numbers according to the order at which they appear in the enumeration of the equivalence structure. Write $[i]_E$ or simply $[i]$ for the i-th class of E. (Classes having a smaller index have a higher "priority".)

Definition 6.3. *An equivalence structure is* eventually bounded *if there is an $n \in \omega$ such that all classes having indices $> n$ are bounded in size by n.*

Note that an eventually bounded structure may have infinite classes or finitely many classes.

Lemma 6.4. For an equivalence structure E, eventual boundedness is a Σ^0_2-property.

Proof. The property says:

$$(\exists n)(\forall i > n)\,\neg\left(\exists a_1, \dots a_{n+1} \in [i] \bigwedge_{i\neq j, i,j\leq n+1} a_i \neq a_j\right).$$

(Recall that the i-th class $[i]$ is not necessarily the class containing the i-th element of E; see the explanation preceding Def. 6.3.) □

6.2.1. *Guessing properness of E.* The preliminary description of the outcomes of this guessing is:

$\Pi^0_2(j)$: E is *not* eventually bounded and the jth class in E is infinite.
Π^0_3: E is proper.
Σ^0_2: E is eventually bounded.

Since an equivalence structure is proper if and only if it is not eventually bounded and does not contain infinite classes, it is clear that the outcomes are exclusive and cover all possible cases.

6.2.2. *Guessing trueness of F. Slicing Σ^0_4 into $(\Pi^0_3(z))_{z\in\omega}$.* Recall that the sentence saying that F is true has complexity Σ^0_4. We represent the respective Σ^0_4-predicate as $\exists z \Pi^0_3(z)$. As usual, we assume that the measured predicates satisfy the property of the uniqueness of existential witnesses. In particular, if $\exists z\Pi^0_3(z)$ holds then there will be exactly one such z.

The outcomes of each $\Pi^0_3(z)$-guessing are:

$\Pi^0_2(j,z)$: This is a Π^0_2 outcome that says that j witnesses the failure of the $\Pi^0_3(z)$ predicate $\forall j \Sigma^0_2(j,z)$.
$\Pi^0_3(z)$: F is true with a Σ^0_4-witness z.

The collection of all $\Pi^0_2(j,z)$-outcomes can be viewed as the $\Sigma^0_3(z)$-complement of $\Pi^0_3(z)$.

6.3. **The strategy for** (F,E,z)**.** Each triple (F,E,z) is associated with a strategy, in which z is interpreted as a potential existential witness for $\exists z\Pi^0_3(z)$ approximating trueness of F. The strategy for one (F,E,z) in isolation relies on the guessing F and E described above, and it also has the following two major tasks.

6.3.1. *The first task: Building H.* The strategy dynamically transforms the computable equivalence structure E into a computable equivalence structure H with the properties:

i. If E is not eventually bounded, and one of the two conditions holds:
(i.1) E has infinite classes, or
(i.2) F looks not true according to $\Sigma^0_3(z)$ (see the previous subsection),
then H has infinitely many classes with almost every class infinite. Furthermore, the number of finite classes in H produced by the strategy associated is specific to the node the strategy and to the outcome of the strategy under which it is produced.
ii. If E is proper and F is true then $H \cong E$.
iii. If E is eventually bounded then H is finite.

Condition *i.* says that H is an infinite junk structure (Def. 4.1). We delay the detailed description of H and the verification of *i*-*iii* until Section 7. Also, a further minor adjustment to this transformation will be introduced in Subsection 8.5 after the tree of strategies $\mathcal{T}$ from [DMN17] is described in sufficient detail. For now, we take these properties for granted.

6.3.2. *The second task: Building $T_H(F)$.* The second task of the strategy is producing $T_H(F)$ based on the dynamic definition of H; here $T_H(F)$ stands for the *modified* version of the Ash-Knight-Oates operator defined in Section 4. As usual, we identify an equivalence structure with the direct sum of cyclic and quasi-cyclic p-groups in which cyclic summands $\mathbb{Z}_{p^n}$ naturally correspond to equivalence classes of size n. According to Lemma 4.2 and assuming the properties *i.-iii.* of H stated in the subsection above, we have the following different scenarios:

a. If H is infinite junk, then $T_H(F) \cong H$.
b. If H is proper, then $(T_H(F))' = F$ and $T_H(F)/(T_H(F))' \cong H$.
c. If H is finite then so is $T_H(F)$.

Furthermore, by Lemma 4.2, the cardinality of the finite $T_H(F)$ in $c.$ can be assumed as large as necessary.

6.4. **Actions of the strategy for (F, E, z).** Whenever the strategy becomes active, it makes one more step in each of the two uniform procedures:

(1) Approximate $T_H(F)$, where H is the uniformly modified version of E satisfying *i-iii* (see Section 7 for details) and $T_H(F)$ is the modified Ash-Knight-Oates operator satisfying *a-c* (see Lemma 4.2) applied to F and H.
(2) Monitor H and guess whether it has infinitely many classes all of which are infinite. Since H is uniformly defined from E, this predicate is uniformly Π^0_2 in (the index for) E. If this predicate fires then the basic module initialises itself by permanently abandoning its current $T_H(F)$. In this case it creates a new version of $T_H(F)$ which is building from scratch. The new version will have a new index in the uniform enumeration of all type $\leq n$ abelian p-groups.

6.5. **The outcomes.** Assuming that H indeed satisfies the claimed properties *i-iii*, the strategy associated with (E, F, z) will have one the following outcomes:

pi_0 This is a Π^0_2 outcome which measures if all classes in H are infinite (and thus there infinitely many such classes).

Every time it is played the strategy is initialised, and its previous version of $T_H(F)$ is abandoned. Recall that the size of the abandoned $T_H(F)$ can be picked as large as necessary, according to Modification 1 from Section 4.

pi_j, $j > 0$: This is a Π^0_2 outcome which says that:
- E is not eventually bounded, i.e., it has arbitrarily large classes of arbitrarily large indices, and
- either the jth class in E is infinite, or F looks not true as witnessed by $\Pi^0_2(j, z)$.

By Lemma 4.2 and assuming properties *i-iii* of H, in this case the strategy produces a computable $T_H(F) \cong H$ which can be identified with G_H composed of at most finitely many cyclic and infinitely many quasi-cyclic direct summands. Furthermore, we will ensure that different strategies always produce non-isomorphic $T_H(F) \cong H$ under their Π^0_2 -outcomes, and also different Π^0_2-outcomes of the same strategy give non-isomorphic $T_H(F) \cong H$. This will be clarified in Section 7. With extra care we will make sure that these infinite junk structures/groups also differ from any infinite junk structure produced by the tree of strategies $\mathcal{T}$ from [DMN17];

see Section 8 for the description of $\mathcal{T}$ and Subsection 6.5 for the abovementioned adjustment.

Π: This is a Π^0_3 outcome that says that E is proper and F is true.

In this case, by Lemma 4.2 and assuming properties *i-iii* of $E \to H$, the strategy outputs a computable basic tree $T_H(F)$ with the properties $(T_H(F))' = F$ and $T_H(F)/(T_H(F))' \cong H \cong E$ (the latter two are identified with the respective groups). Furthermore, since $E \cong H$ is proper and F is true of type $< n$, the Ulm type of $T_H(F)$ is at least 2 and at most n.

fin: This is a Σ^0_2-outcome which says that E is eventually bounded.

In this case $T_H(F)$ is finite. Furthermore, its cardinality can be controlled and made arbitrarily large, if necessary, according to Modification 1.

To finalise the description of the basic strategy we must give a detailed description of the transformation $E \to H$ and verify its claimed properties.

7. The description of $E \to H$

First, in Subsection 7.1 we describe a transformation $E \to \widehat{H}$ which takes care of most properties *i-iii* with the exception of the "furthermore" part of *iii*. Then in Subsection 7.2 we further adjust $\widehat{H}$ and describe a transformation $\widehat{H} \to H$ which is based on a strategy from [DMN17] and which also gives property *iii* in full. This modification is highly convenient in the general case of many strategies working together. Compared to the rest of the paper the content of this section is rather elementary.

7.1. **The definition of $\widehat{H}$.**

Given an infinite computable equivalence structure E, the strategy produces a computable equivalence structure $\widehat{H}$ with the properties:

i. If E is not eventually bounded (Def. 6.3), and one of the two conditions holds
 (i.1) E has infinite classes, or
 (i.2) F looks not true according to $\Pi^0_2(j, z)$ (see 6.2.2),

then $\widehat{H}$ has infinitely many classes with almost every class infinite.

ii. If E is proper and F is true then $\widehat{H} \cong E$.

iii. If E is eventually bounded then $\widehat{H}$ is finite.

We write $[m]_L$ for a class of an equivalence structure L with index m. Say that a stage s is expansionary if the parameter $\max\{card[i]_{E_s}, i \leq s\}$ has increased from the previous expansionary stage s'. The parameter measures whether the structure E has arbitrarily large classes with arbitrarily large indices. The simple construction below acts only at expansionary stages.

7.1.1. *Construction.* At every stage, each class in $\widehat{H}_s$ is matched with a class in E_s. Suppose at a stage $[n]_{\widehat{H}}$ is copying $[i]_E$. If $[i]_E$ has grown in E or the ith Π^0_2 instance of the Σ^0_3 predicate "F is not true (z)" has fired, then perform the following action. *Initialise* each class $[k]$ in $\widehat{H}$ that satisfies

(1) $k > n$, and
(2) $[k]_{\widehat{H}}$ has been copying a class $[j]_E$ with $j \geq i$.

Each initialised class grows by one extra element and will be assigned to some large enough new class in E (if it exists). Until such large enough classes are found the whole strategy (not just this simple procedure describing $\widehat{H}$) ceases its action. Then, once large enough classes are found, each currently abandoned classes of E is assigned to a new class in $\widehat{H}$. This ends the construction.

7.1.2. *The verification of i, ii and iii.* To see why *iii* holds, recall that the procedure constructing $\widehat{H}$ acts only at expansionary stages. Since there are only finitely many such, $\widehat{H}$ remains finite. To check *i* and *ii*, note that each initialised class must grow. A class can be initialised only due to some larger index class growing or due to some higher priority Π^0_2 instance of the predicate "F is not true (z)" firing; furthermore, in the former case this larger $\widehat{H}$-index class must be copying a larger E-index class. There are only finitely many such. Thus, if all classes in E are finite and F looks not true according to instance z, then each class can be initialised only finitely often. Also, a class in E has to change its clone in $\widehat{H}$ only if a class with a smaller E-index grows. Therefore, *ii* follows by induction. To check *i*, assume that $[j]$ is the left-most class of E – i.e., the one with the smallest index – that grows to infinity. Since all classes to the left of it are finite, there is a stage after which the class is stably assigned to a clone in $\widehat{H}$, call this clone $[k]$. There exist at most finitely many classes of $\widehat{H}$ to the right of $[k]$ that are controlled by classes in E having index less than the index of $[j]$. All the rest are initialised infinitely often. Since E has arbitrary large classes with arbitrary big indices, every search for a new appropriate image for an initialised class is successful. In particular, E has infinitely many classes, and therefore so does $\widehat{H}$. Since each initialised class must grow, co-finitely many classes of $\widehat{H}$ are infinite.

7.2. **The transformation from $\widehat{H}$ to H.** Fix a uniformly computable collection of non-intersecting intervals $I_0, I_1, \ldots, I_n \ldots$ in ω which form its full partition, where the smallest number of I_n is equal to the largest number of I_{n-1} plus 1. We write $\max I_n$ for the largest number of I_n. (In the construction we will also make sure that $\max I^\sigma_i \neq \max I^\tau_k$ for $\sigma \neq \tau$ and any strictly positive $i, k \in \omega$.)

We are given $\widehat{H}$ which is either proper, or is an infinite junk, or is finite (cf. *i*-*iii*). Recall that, according to our convention, every class of $\widehat{H}$ receives an index according to the stage at which it appears in the enumeration of $\widehat{H}$. The uniform definition of $\widehat{H}$ contained in the subsection above has the following property. If the size of $[i]$ in $\widehat{H}$ is infinite and has infinitely many classes, then so is $[k]$ for each class $[k]$ having its index larger than the index for $[i]$. We must uniformly build a computable equivalence structure H and a map $\psi : H \to \widehat{H}$ by stages.

The idea is rather simple. We construct H so that it copies $\widehat{H}$, but the isomorphism $\psi : C \to \widehat{H}$ is defined not class-by-class but block-by-block. If some class in the k-th block of $\widehat{H}$ has grown, then in H initialise all ψ-preimages of j-blocks for $j \geq k$. Whenever we initialise a block in H each class in the block is increased in size.

We give formal details. At stage s, if class of $\widehat{H}$ having index $j \in I_k$ has grown in size, then:

(1) Declare ψ undefined for every class of $\widehat{H}_s$ having its index in I_m for some $m \geq k$.
(2) Grow all classes of H_s which currently have no ϕ-image to a size greater than any number mentioned so far.
(3) Speed up the enumeration of $\widehat{H}$ and search for new, larger images for the finitely many classes in H_s for which ψ is currently undefined.
(4) If (2) is ever finished, introduce a new classes in H_s and match them with those classes of $\widehat{H}$ which currently have no ψ-preimages. Goto (1).

Lemma 7.1.
(1) *If $\widehat{H}$ is proper then $H \cong \widehat{H}$.*
(2) *If $\widehat{H}$ is finite then H is finite too.*
(3) *If $\widehat{H}$ is an infinite junk then so is H. Furthermore, either H has all classes infinite or the total number of finite classes in H is equal to $\max I_k$ for some k.*

Proof. (1). By induction on the index i of a class $[k]_{\widehat{H}}$ and the index m of the block I_m such that $i \in I_m$, every class $[k]$ in $\widehat{H}$ eventually finds a stable ψ-preimage in H. Thus, in this case ψ is a Δ^0_2-isomorphism of equivalence structures witnessing $H \cong \widehat{H}$.

(2). This is obvious.

(3). Let m be the smallest such that there is an infinite class in $\widehat{H}$ having index $j \in I_m$. Then the only classes which have stable ψ-preimages in H are the classes whose indices are in I_n for some $n < m$. If a class in H does not have a stable ψ-image then its size is driven to infinity; indeed, since $\widehat{H}$ is an infinite junk the search at (3) of the procedure describing H is always successful, and according to (2) whenever ψ is redefined the class must be grown. If $m = 0$ then all classes in H end up infinite, otherwise let $k = m$-1. □

The lemma above and the properties of $E \to \widehat{H}$ imply that the uniform transformation $E \to \widehat{H} \to H$ satisfies *i*-*iii* from 6.3.1, as desired. One further insignificant restriction to the choice of intervals I_m will be explained in Subsection 6.5.

8. The tree of strategies for Ulm type 1 groups

The construction in [DMN17] consists of the tree of strategies, the junk collector, and also an external and independent module enumerating all equivalence structures having finitely many classes at least one of which is infinite and equivalence structures which have infinitely many classes and are eventually bounded. In this section we describe the tree of strategies from [DMN17] with the detail sufficient for our purposes; the junk collectors will be discussed in the next section.

The tree of strategies from [DMN17] is used without any significant modification, i.e., it can be essentially literally copied from [DMN17]. The tree and various strategies associated with its nodes act independently from the rest of the construction, and the only interaction with the rest of the construction is via the junk collector. And even then this interaction is literally the same as in the proof of [DMN17]. All we need to do is:

(1) interpret equivalence structures as the respective Ulm type 1 groups, and

(2) for every strategy associated with some σ along the tree, the infinite junk structures potentially produced by σ are non-isomorphic to any infinite junk structure produced by a strategy for (F_i, E_j, z) or by any other $\tau \neq \sigma$.

The first assertion is just a triviality, and the second is not really a modification either, for the original construction in [DMN17] already ensured that different nodes and different outcomes produce non-isomorphic infinite junk structures, and the precomputed bounds on the number of exceptional classes can be kept exactly the same as in [DMN17]. We will elaborate on this point at the very end of this section, where specifics will be spelled out.

No further adjustment is necessary. Thus, if the reader is familiar with [DMN17] they can skip the rest of the section which is devoted to a compressed description of the tree of strategies from [DMN17], the strategies associated with its nodes, and of the types of equivalence structures produced under different outcomes. We start with an idea. (We adjust the notation from [DMN17] to avoid conflicts with the notation in the present article.)

8.1. **Idea.** Let X_i be the i-th equivalence structure in their natural uniform enumeration with repetition, in which the k-th class of the i-th structure is represented by the k-th column of the computably enumerable set W_i. To produce a Friedberg enumeration of all isomorphism types of computable equivalence structures, we could (naively) start off by declaring X_0 be the first in the list. To decide whether X_1 must be put into the list, we must see if $X_0 \cong X_1$. The relation $X_i \cong X_j$ is Π^0_4-complete, but X_1 must be placed into the Friedberg list only when $X_0 \not\cong X_1$ which is Σ^0_4.

We spread this Σ^0_4-guessing over infinitely many Π^0_3-nodes in the tree of strategies, with each node working with its own existential witness z for a given X_i which approximates whether $X_i \cong X_j$ for some $j < i$. Each node working with (i, z) dynamically replaces its structure $X_i = X$ with a structure U using a uniform transformation similar to that from Section 7. In this transformation, if the structure has arbitrarily large finite classes (which is a Π^0_3 condition) then the output structure U is isomorphic to X. Otherwise we end up with either a finite X whose size can be assumed as large as necessary, or an infinite junk having the number of exceptional finite classes taken from a computable set specific to the strategy.

8.2. **The basic strategy.** Each basic strategy is associated with a pair (i, z). It monitors the i-th equivalence structure X_i and approximates the Π^0_3-instance of the Σ^0_4-predicate measuring $\Xi(X_i) \wedge (\forall k < i) X_i \not\cong X_k$, where $\Xi(X_i)$ is the Π^0_3 predicate saying that X_i has arbitrarily large finite classes. Let $P(z, i)$ be the $\Pi^0_3(z)$ predicate such that $\Xi(X_i) = \exists z P(z, i)$. As usual, without loss of generality $\exists z P(z, i)$ implies that there exists exactly one such z.

The strategy dynamically transforms X_i into an equivalence structure $U = U_i$ with the properties:

(1) If $P(z, i)$ holds then $U \cong X$.
(2) If $P(z, i)$ fails then either U is finite or U is infinite junk with the number of exceptional finite classes coming from a computable set specific to the strategy.

The transformation is similar to the one described in Section 7. A rigorous description of this transformation and its verification are contained in Subsections 2.3.2-2.3.4 and Lemmas 2.1 and 2.2 of [DMN17], but in a different notation. (In the notation of [DMN17], the input structure is denoted by E_τ, the output structure is U_τ, and the partial isomorphism between the two at every stage is ℓ_τ. The description of the transformation in incorporated into the description of the basic strategy.) The following outcomes are possible (see 2.3.5 of [DMN17]):

- The Σ^0_2 outcome *wait*. It is played when either X_i has only finitely many classes or it is eventually bounded (i.e., does not have arbitrarily large classes of arbitrary large index). In this case we may assume that the size of some class of U is as large is necessary on the stage when U is defined.
- The Π^0_2 outcome *init*. If this is the true outcome then X_i has infinitely many classes but too few of them are finite, namely less than k finite classes, where k is specific to each strategy. This is clearly a uniformly Π^0_2 condition. The outcome is played if the Π^0_2 predicate fires, and in this case the strategy initialises itself. More specifically, the current equivalence structure U built by the strategy is permanently abandoned and the strategy starts building a new equivalence structure which will have a large index in the global Friedberg enumeration.
- The Π^0_2 outcome pi_j. This is the j-th instance of the Σ^0_3-predicate saying that $P(z, i)$ fails; in other words, either X_i *could be* isomorphic to some X_j, $j < k$, or it does not have arbitrarily large finite classes. In this case U is an infinite junk structure. Whenever the outcome is played again it comes with the best current approximation c to the number of finite exceptional classes. This number k of exceptional finite classes of U is necessarily taken from a computable parameter set specific to the strategy and the parameter j of this outcome. Different parameter sets for different strategies do not overlap. If this outcome is the true outcome then there exists a stage s such that $c[t] = c[s]$ for every $t \geq s$ and $c[t]$ is correct.
- The Π^0_3 outcome pi_3. It says that $P(i, z)$ holds, and thus $E_i \not\cong E_k$ for any $k < i$, and also E_i contains arbitrarily large finite classes. Under this outcome the strategy produces $U_i \cong E_i$.

8.3. **The tree of strategies, the current true path, initialisation.** The order of the outcomes is:

$$init < pi_0 < pi_1 < \ldots < pi_3 < wait.$$

The tree $\mathcal{T}$ is composed according to this order; of course under the pi_3 outcome measuring $P(i, z)$ there is no other node working with some $z' > z$ in $P(i, z')$.

The definition of the current true path is standard for such constructions with explicit Π^0_3-outcomes, with the pi_3-outcome visited in-between pi_j outcomes, so that the true path is $0'''$-computable. No links or scouting reports or other tricks peculiar to some $0'''$ proofs are necessary. The definition of initialisation is not entirely standard; the only not entirely standard part being that the nodes below the pi_3 outcome of σ are forced to play their pi_2j outcomes if σ plays its pi_2j outcome.

We note that in [DMN17] there was an unnecessarily complex resolution of the possibility of several Π^0_2-outcomes of the same τ played infinitely often; this difficulty can be resolved entirely and elementarily by using the uniqueness of $\exists$-witnesses throughout. We must of course dynamically adjust the definitions of all outcomes of τ (including those played if τ is off the current true path) depending on the position of τ on the tree. In [DMN17], we defined a dynamic explicit version of the above-mentioned transformation, but of course it did not have to be explicit.

8.4. **Structures produced by $\mathcal{T}$.** We will not sketch the verification; see [DMN17] for the details of the proof. We take correctness of the tree-construction sketched above for granted. If we view the tree of strategies $\mathcal{T}$ as one large module, its cumulative products can be classified as follows:

- *Equivalence structures having arbitrarily large finite classes.* All such structures are enumerated under the pi_3-outcomes along the true path, and without repetition (up to isomorphism).
- *Finite structures.* These come from true *init*- and *wait*-outcomes of various nodes in the tree, and also are produced due to initialisation. By making them larger than any number seen so far in the construction (see, e.g., Modification 1), we ensure there is no repetition among them, but we do not guarantee that all finite structures are produced by the tree.
- *Infinite junk structures produced by true pi_j-outcomes of various structures.* Note that some strategies off the true path can be forced to play their pi_j-outcomes. The number of sizes of exceptional classes is different for different nodes and below different outcomes of the same node. At every stage the isomorphism type of the structure is guessed, with the guess eventually becoming correct if the outcome is played infinitely often.

  More specifically, if m is the number of times the strategy (call it τ) has been initialised, then the number of finite classes in U_τ produced under the true outcome $pi_2 j$ of τ should be between $\langle \tau, m, j\rangle$ and $2\langle \tau, m, j\rangle$, where the standard pairing function $\langle i, j\rangle$ is replaced with $3^{\langle i,j\rangle}$ (see the very end of 2.3.2 of [DMN17] for this convention); this is Lemma 2.1 of [DMN17]. These parameters are highly flexible, allowing us to change the base of the exponent and the exact choice of enumeration of τ. But the approach in [DMN17] already gives sufficiently sparsely distributed intervals, so no further adjustment will be necessary.

8.5. **The complete separation of infinite junk structures.** Now, since we have explained the role of the intervals $[\langle \tau, m, j\rangle, 2\langle \tau, m, j\rangle]$, we are ready to introduce the following elementary but important adjustment to the basic strategy from Section 6.

Modification 2. *We assume that for every strategy σ working with some (F_i, E_j, z), the parameters $\max I_k^\sigma = \max I_k$ described in Subsection 7.2 are taken from the complement of the set*

$$\bigcup_{\tau\in\mathcal{T}} [\langle \tau, m, j\rangle, 2\langle \tau, m, j\rangle],$$

where $\mathcal{T}$ is the tree of strategies from [DMN17]. *We furthermore assume that $\max I_k^\sigma \neq \max I_j^{\sigma'}$ whenever either $\sigma \neq \sigma'$ or $k \neq j$.*

Infinite junk structures produced by various Π^0_2-outcomes of different strategies are non-isomorphic. Thus, there is no conflict between Π^0_2-outcomes of different strategies, regardless of whether they live on the tree $\mathcal{T}$ or work with some triple (F_i, E_j, z). Any two distinct Π^0_2-outcomes of the same strategy (on the tree or working with a triple) produce non-isomorphic infinite junk structures as well.

Informally, each Π^0_2 outcome will "know" the isomorphism type of the structure it will produce. Since we assume uniqueness of existential witnesses throughout, the true Π^0_2-outcome is the only one which guesses the isomorphism type of the infinite junk structure correctly infinitely often. More formally, if an infinite junk structure L is produced under a true Π^0_2-outcome of some $\tau \in \mathbb{T}$ or some σ working with (F_i, E_j, z), then L comes together with a computable sequence $(l_s)_{s\in\omega}$ such that the unique number l mentioned in the sequence $(l_s)_{s\in\omega}$ infinitely often describes the sizes of the finitely many exceptional classes in L. (If L is not infinite junk then $(l_s)_{s\in\omega}$ has no such l.) In the case of $\tau \in \mathbb{T}$ the finite parameter comes from the dynamic definition of the respective interval $[\langle\tau, m, j\rangle, 2\langle\tau, m, j\rangle]$, and in the case of σ working with (F_i, E_j, z) this parameter is the current $\max I^\sigma_m$, where m corresponds to the outcome.

9. The junk collector

The junk collector can be extracted from [DMN17] without any further modification. For completeness, we explain the action of the junk collector in fairly complete detail.

We call a structure *a junk structure* if it is either a finite structure or an infinite junk structure produced by one of the strategies. Junk structures can be of two different kinds:

(1) Finite junk. These are finite abelian p-groups/equivalence structures which are either produced due to initialisation or are built if the Σ^0_2-outcome is the true outcome. Because of Modification 1, the cardinalities of these finite groups may be assumed to be large and unseen at the stage when they are first introduced; see Lemma 4.2(3).

(2) Infinite junk (see Def. 4.1). These are produced under various Π^0_2-outcomes, which are not their left-most Π^0_2-outcomes, of basic strategies either working with (F_i, E_j, z) or along the tree $\mathcal{T}$. According to Modification 2 in the preceding Subsection 8.5, the isomorphism type of the infinite junk structure produced by $\sigma\widehat{\ }\xi$, where ξ is the Π^0_2-outcome of σ played infinitely often, will be uniquely determined by σ and ξ, regardless of the type of the strategy σ. At every stage at which the outcome is played we will also have the current best guess on the isomorphism type of the structure.

The junk collector consists of two submodules working in coordination with each other.

9.1. **The infinite junk collector.** The task of this global strategy is to ensure that each isomorphism type of infinite junk structure H is represented in the global enumeration, and exactly once. Here the isomorphism type of an infinite junk structure is identified with the isomorphism type of the respective abelian p-group of Ulm type 1. Note that under each Π^0_2-outcome, which is not the left-most outcome, we have a specific guess on the isomorphism type of the infinite junk produced by the respective strategy. Call this isomorphism type L. If the outcome is played again at stage s, then we say that L is *active* at s. Otherwise, we say that it is *not active* at the stage. We assume that at most one structure is active at a given stage.

9.1.1. *Idea.* Initiate an enumeration of all isomorphism types of infinite junk structures. If L becomes active at a later stage, then we are in the danger of having repetitions, for the following reason. Suppose an enumeration of $L' \cong L$ has already been initiated by the infinite junk collector. When L becomes active, we stop building L' and permanently put the currently finite L' into the finite junk collector (to be explained). We also artificially adjoin a very large cyclic summand to L' to make it look different from all the other finite structures that we have ever seen in the construction so far. While L is no longer active, we re-introduce its isomorphism type to the junk collector by using a large $L'' \cong L$.

9.1.2. *The formal description.* We define a computable sequence $(J_i^s)_{s,i\in\omega}$ of infinite junk equivalence structures. The sequence $(J_i^s)_{s,i\in\omega}$ can be thought of as a computable map ν which on input (i, s) outputs the index of some computable abelian p-group. At every stage we will of course have only a finite part of each of these structures in their natural uniform enumeration.

The input of the infinite junk submodule is a uniform enumeration of abelian p-groups some of which can be infinite junk. We write $L_0, L_1, \dots$ to denote these groups. At every stage each L_i is finite and is identified with the respective equivalence structure E_{L_i} with all possible uniformity; see Proposition 2.1. This list is uniformly produced by sub-strategies of the main strategy and the tree $\mathcal{T}$ all working together, but the exact nature of this list is not important. We need only the following assumptions about this list.

(a1): We identify each L_i with its index which is uniformly computable from i; without loss of generality we may assume that the complement of the set of all these indices is an infinite computable set.

(a2): At every stage at most one such $L = L_i$ can be declared *active* which means that, in a Π_2^0-fashion, we have more evidence that L may end up being an infinite junk structure. In this case the intended isomorphism type of L is also given in the form of a finite parameter describing the exceptional finite classes of L. At such a stage L grows in size to a very large cardinality. If L is active infinitely often then this parameter is the only one which appears as the best current guess infinitely many times (cf. Subsection 8.5).

(a3): Also, if $L_i \neq L_j$ then their parameters from (a2) above never describe the same isomorphism type of an infinite junk structure (cf. Modification 2).

At stage 0, initiate a uniform enumeration $(J_i^0)_{i\in\omega}$ of all isomorphism types of abelian p-groups J such that the respective E_J is an infinite junk equivalence structure. Each isomorphism type comes with a (strong) index describing the finitely many exceptional classes in the respective J_i^0. Unless interrupted and declared abandoned (to be defined), each J_i^0 eventually ends up isomorphic to the infinite junk structure with the declared finite description.

As stage s, consider the following cases:

- Some L is active at the stage. Let i be the unique index such that, according to the parameter describing L (see (a2)), we should have $L \cong J_i^0$. Set $J_i^{s+1} = L$ and declare J_i^s *abandoned* and place it into the finite junk collector. Adjoin a very large class to the structure before permanently abandoning it.
- No L is active. If L was active at stage s-1 and currently $J_i^s = L$, then introduce a new D having a large index (see (a1)) which, unless interrupted and declared abandoned, will have the same isomorphism type as the intended isomorphism type J_i^0 at the end of the construction; define $J_i^{s+1} = D$.

In any case make one more step in the approximation of each J_i^s, and go to the next stage.

Now to the verification. We identify equivalence structures with their computable indices and with the respective groups. Let $\underline{\lim}_s J_i^s$ be equal to the structure X having the smallest index such that there exists infinitely many stages s at which $J_i^s = X$.

Lemma 9.1. *Suppose there exist infinitely many stages at which L becomes active, and let i be such that $J_i^0 \cong L$. Then $J_i = \underline{\lim}_s J_i^s = L$.*

Proof. At every stage s at which L becomes active we set $J_i^{s+1} = L$. Furthermore, if some other L' becomes active at a stage t, then we set $J_j^{t+1} = L$ for some $j \neq i$, because the finite parameter describing L' corresponds to a non-isomorphic infinite junk structure by (a3). Also, every time L becomes active, the structure J_i^s is declared abandoned and will never be set equal to J_i^t at any later stage t. It follows that $\underline{\lim}_s J_i^s = L$. □

Lemma 9.2. *Suppose X is an infinite junk structure such that there is no $L \cong X$ in the input list which becomes active at infinitely many stages. Then for some i, $J_i = \underline{\lim}_s J_i^s \cong X$.*

Proof. Let i be such that the intended isomorphism type of J_i^0 is the same as the isomorphism type of X. By (a3) combined with (a2), there will be at most one L in the list which could potentially be isomorphic to X in the limit. Thus, there are at most finitely many stages s at which $J_i^s \neq J_i^{s+1}$. Let s' be least such that $J_i^s = J_i^{s+1}$ for each $s > s'$. If $s' = 0$ then $\underline{\lim}_s J_i^s = J_i^0$, the enumeration of J_i^0 is never interrupted, and thus we end up with $X \cong J_i^0 = \underline{\lim}_s J_i^s$ by the choice of i. Otherwise, $\underline{\lim}_s J_i^s = D$ for some D picked at s' and which, unless interrupted, has the same isomorphism type as the intended isomorphism type J_i^0. Since $J_i^{s'+1} = D$ will never be declared abandoned, and since D is described by the same finite parameter as was initially picked for J_i^0, the choice of i implies $X \cong D = \underline{\lim}_s J_i^s$. □

As before, let $J_i = \underline{\lim}_s J_i^s$.

Lemma 9.3. *The sequence $(J_i)_{i\in\omega}$ mentions each isomorphism type of infinite junk structures/groups exactly once.*

Proof. Fix some isomorphism type X. Suppose there is a member $L = L_i$ of the input list which becomes active at infinitely many stages and such that $L \cong X$. Then for some i we will have $J_i = L \cong X$. No other J_j with $j \neq i$ can be isomorphic to X. On the other hand, if no such L_i exists then Lemma 9.2 implies that for some i we have $J_i = D \cong X$, and again for exactly one such i. □

Define a uniform enumeration $(Z_i)_{i\in\omega}$ which includes all of the $(L_i)_{i\in\omega}$ and further expand it by structures which are of the following two sorts:

- A finite structure expanding some J_i^s which was declared abandoned at stage s;
- An infinite junk structure $J_i = \underline{\lim}_s J_i^s$ which, for some s', will have index $J_i^{s'}$ (cf. Lemma 9.2).

In particular, the first clause will cover the case when $J_i = \underline{\lim}_s J_i^s$ is undefined and thus for every s the index J_i^s is eventually abandoned. It also covers the case when the monitored structure of isomorphism type J_i^0 already appears in the list $(L_i)_{i\in\omega}$. The second clause covers the case when the isomorphism type of J_i^0 is not mentioned among $(L_i)_{i\in\omega}$. With a bit of extra work, the lemmas above imply the following:

Proposition 9.4. *Given a uniform list $(L_i)_{i\in\omega}$ with properties $(a1)$-$(a3)$ and which is injective on isomorphism types, the finite junk collector outputs a uniform list $(Z_i)_{i\in\omega}$ which mentions each member of $(L_i)_{i\in\omega}$ exactly once and mentions each isomorphism type of infinite junk structures exactly once. In addition to all infinite junk structures and members of $(L_i)_{i\in\omega}$, it may only contain some isomorphism types of finite structures/groups, and also without repetition.*

Proof. Most of the work has already been done, it remains to check the claimed properties related to finite structures. It is clear that the extra isomorphism types may come from J_i^s only. But by assumption (a2) they are all distinct, for at the stage at which they are introduced they are larger than any other finite isomorphism mentioned so far. □

This list $(Z_i)_{i\in\omega}$ will serve as the input for the finite junk collector which is described below.

9.2. **The finite junk collector.**

This global strategy must ensure that every isomorphism type of a finite abelian p-group is represented in the enumeration. As usual, we can identify such groups with finite equivalence relations. Recall also Modification 1.

9.2.1. *Idea.* We initially start with a uniform enumeration of all finite abelian p-groups. At every stage we have only finitely many of them already listed, and some of these finitely many groups may have to be further expanded to a larger finite group, but only at most once. This is because new finite abelian groups appear in the construction only due to the actions of the main module, the tree $\mathcal{T}$, and the infinite junk collector. Such groups could be of three different kinds:

(1) Finite abelian groups that are permanently abandoned by a node in $\mathcal{T}$ or a module working with a triple (F_i, E_j, z) due to initialisation. According to Modification 1 we adopted in the definition of $T_H(F)$, if a strategy is initialised then its structure gets a very large simple chain. Similarly, if an equivalence structure is abandoned by a strategy in $\mathcal{T}$ then we adjoin a very large new class to it. This makes the isomorphism type of the abandoned finite group/equivalence structure unique at the respective stage.
(2) Finite abelian groups that are permanently abandoned by the infinite junk collector. The isomorphism type of this abandoned finite group is unique at the stage, because a very large finite class/summand is adjoined to it; see the description of the infinite junk collector.
(3) Structures that are finite approximations to a $T_H(F)$ of some τ at a finite stage. According to Modification 1, at every stage s at which some progress has been made in the approximation of the respective $T_H(F)$, the finite structure $T_H(F)[s]$ has its isomorphism type unseen so far in the construction. This is achieved by artificially adjoining a very long simple chain

to the root of the Π^0_2 p-basic tree every time the strategy has to temporarily stop enumerating its $T_H(F)$.

If a group A of a sort (1), (2), or (3) is permanently put into the finite junk collector, then there is no isomorphic finite group already listed by the finite junk collector by this stage. This is because the isomorphism type of the new group is artificially made very large in each of the three cases. It is thus routine to make sure that these groups are incorporated into the enumeration, and no finite abelian p-group isomorphic to these will be ever introduced by the finite junk collector.

However, in case (3) the group may resume growing, and we reintroduce the finite isomorphism type that it had just before it grew again. This is done using a new finite group with a large index. We permanently put this new group into the enumeration. This group will never become isomorphic to any other finite group in the construction, because all groups produced by other strategies at later stages will have cyclic summands that are too big, and because the finite junk collector itself will never duplicate the finite group at any later stage.

9.2.2. *The formal description.* The input is a uniform enumeration $(Z_i)_{i\in\omega}$ of abelian p-groups. At every stage each finite $Z_i[s]$ is identified with the respective equivalence structure $E_{Z_i[s]}$, with all possible uniformity. In the construction this list is produced collectively by $\mathcal{T}$, sub-modules of the main module, and the infinite junk collector. We will need only the following dynamic properties of this list. These properties are immediate consequences of Modification 1 and the analysis contained in Subsection 8.4.

(b1): At every stage s there is at most one i for which $Z_i[s+1]$ is larger than $Z_i[s]$, in this case we also assume that the cardinality of $Z_i[s+1]$ is larger than any number mentioned so far in the construction. This applies to the case when $Z_i[s+1]$ is newly introduced too.

(b2) At every stage s the finite list $(Z_i[s])_{i\leq s}$ contains no repetition up to isomorphism.

The finite junk collector defines its own sequence $(D_i)_{i\in\omega}$ by stages, as follows.

At stage 0, set D_0 equal to the empty equivalence structure/the trivial group.

At stage s, define D_s be equal to the index of the least isomorphism type (with respect to the natural uniform enumeration of such types) which is currently not mentioned among $(Z_k[s])_{k\leq s}$ and D_j, $j<s$. Go to the next stage.

Expand the enumeration $(Z_i)_{i\in\omega}$ by uniformly adjoining all structures $(D_i)_{i\in\omega}$ to this enumeration. Let $(B_i)_{i\in\omega}$ be the resulting combined uniform enumeration.

Lemma 9.5. *For every i and j, $Z_i \ncong D_j$.*

Proof. When D_j is defined at stage j it is set equal to a finite structure/group not isomorphic to any group mentioned so far in the enumeration. By property (b1), no $Z_i[t]$ with $i\leq j$ can be set equal to a finite structure isomorphic to D_j since its cardinality is larger than the cardinality of D_j. For $i>j$, the structure $Z_i[t]$ is very large when it is introduced, and thus it cannot be isomorphic to D_j. □

Lemma 9.6. *For any $i\neq j$, $D_i \ncong D_j$.*

Proof. According to the instructions at stage j, no D_i with $i < j$ could be isomorphic to D_j. Similarly, according to the instructions at stage i, D_i with $i > j$ cannot be isomorphic to D_j. □

Lemma 9.7. *Every isomorphism type of finite equivalence structures/finite abelian p-groups is mentioned in $(B_i)_{i\in\omega}$ exactly once.*

Proof. Each isomorphic type is mentioned at least once by the choice of D_s at stage s. Indeed, assuming it is not mentioned, we would arrive at a contradiction, for eventually all newly introduced members of $(Z_i)_{i\in\omega}$ will be either stable or will be too large, and thus for some s we will have to fill in the gap using D_s. By (b1) and the two lemmas above, the enumeration is injective on finite isomorphism types. □

The lemmas imply the following:

Proposition 9.8. *On input of a uniform enumeration $(Z_i)_{i\in\omega}$ which is injective on isomorphism types and satisfies (b1)-(b2), the finite junk collector produces a uniform enumeration $(B_i)_{i\in\omega}$ which is injective on isomorphism types and mentions each isomorphism type from $(Z_i)_{i\in\omega}$ and each finite isomorphism type.*

This finishes the description of the finite junk collector.

10. The construction

We are ready to put all the essential components together. The construction consists of three phases. The output of the first phase is the input of the second phase, and the output of the second is the input of the third. Apart from this obvious correlation via the input/output, there is no further interaction between the three phases.

10.1. **Phase 1.**

Fix the effective enumeration $(F_i, E_j, z)_{i,j,z\in\omega}$ which was defined at the beginning of Section 5. Also, fix the tree of strategies $\mathcal{T}$ defined in Section 8 and the strategies associated with its nodes.

At the first phase we let all the basic strategies associated with each triple (F_i, E_j, z) and the strategies associated with $\mathcal{T}$ act according to their instructions; the instructions can be found in Section 6 and Section 8, respectively.

Working together, these strategies produce a uniform enumeration $(L_i)_{i\in\omega}$ of computable abelian groups which, as we shall argue, satisfy conditions $(a1)$-$(a3)$ from Subsection 9.1.2.

10.2. **Phase 2.**

On input the enumeration $(L_i)_{i\in\omega}$ listed at Phase 1, let the infinite junk collector act according to its instructions, as described in Subsection 9.1. Let $(Z_i)_{i\in\omega}$ be the uniform enumeration produced as the result of these actions. We will argue that $(Z_i)_{i\in\omega}$ will satisfy conditions $(b1)$-$(b2)$ from Subsection 9.2.

10.3. **Phase 3.**

On input $(Z_i)_{i\in\omega}$, let the finite junk collector act according to its instructions and produce a uniform enumeration $(B_i)_{i\in\omega}$.

Finally, fix some Friedberg enumeration $(M_i)_{i\in\omega}$ of all abelian p-groups of Ulm type 1 which correspond to equivalence structures having finitely many classes at least one of which is infinite and to eventually bounded equivalence structures having infinitely many classes. Merge $(B_i)_{i\in\omega}$ with $(M_i)_{i\in\omega}$ to produce an enumeration $(C_i)_{i\in\omega}$. (For $i = 0, 1, \ldots$, set $C_{2i+1} = B_i$ and $C_{2i} = M_i$.)

We will argue that $(C_i)_{i\in\omega}$ is a Friedberg enumeration of all computable abelian groups of Ulm type $\leq n$.

10.4. **Verification.** As usual, we identify equivalence structures and the respective Ulm type 1 groups throughout.

Lemma 10.1. *The enumeration $(L_i)_{i\in\omega}$ produced at Phase 1 has the following properties:*

(1) *It contains no repetition, up to isomorphism.*
(2) *It includes all isomorphism types of computable abelian p-groups of Ulm types m, $1 < m < n$.*
(3) *It mentions each isomorphism type of computable abelian p-groups having Ulm type 1 in which there are arbitrarily large finite cyclic summands.*
(4) *It mentions some abelian p-groups of Ulm type 1 corresponding to infinite junk structures. In each of these cases the respective group L_i in the list comes with an eventually stable sequence $(l^i_s)_{s\in\omega}$ such that the number $l^i = \lim_s l^i_s$ describes the sizes of the finitely many exceptional classes in E_{L_i}. (If L_i is not infinite junk then $(l^i_s)_{s\in\omega}$ will be divergent.)*
(5) *It includes some finite abelian p-groups.*
(6) *Apart from the isomorphism types described in* (2)-(4)*, no further isomorphism types will be enumerated.*
(7) *It satisfies* $(a1)$-$(a3)$ *from 9.1.2.*

Proof. (2): As we argued in Section 5, every computable abelian p-group A must have A' true and A/A' proper. Theorem 3.7 implies that, for some pair (F_i, E_j) we will have $F_i \cong A'$ and $A/A' \cong E_j$. The Σ^0_4-predicate described in Section 6 holds for this pair. In particular, for exactly one z the basic strategy working with (F_i, E_j, z) has a true Π^0_3-outcome; see Subsection 6.5 for the detailed analysis of the outcomes. Under this outcome the strategy produces $T_{H_j}(F_i) \cong A$.

(3): See Subsection 8.4 for a detailed analysis of the structures produced by $\mathcal{T}$.

(4): This is explained in Subsection 8.5.

(5): This is merely an observation based on the descriptions of the strategies.

(6): This follows from the detailed analysis of the outcomes contained in Subsections 6.5 and 8.4.

(7): Condition $(a1)$ is a triviality, $(a2)$ is reformulation of (4) of this lemma, and $(a3)$ is Modification 2 in Subsection 8.5.

(1): We use the same notation as in the proof of (2) of this lemma. Since the enumerations $(F_i)_{i\in\omega}$ and $(E_j)_{j\in\omega}$ are Friedberg and since we assumed uniqueness of existential witnesses throughout, the groups produced under true Π^0_3-outcomes corresponding to different pairs (F_i, E_j) are non-isomorphic, and there is at most one true Π^0_3-outcome for each such pair. The true Π^0_3-outcomes of strategies along the true path of $\mathcal{T}$ witness that the construction produces a complete list of all computable equivalence structures having arbitrarily large finite classes; see Subsection 8.4.

Modification 2 and the analysis contained in Subsection 8.5 implies that infinite junk structures produced by different strategies cannot be isomorphic. Finally, Modification 1 in the proof of Proposition 2.6 and the analysis contained in Subsection 8.4 guarantee that finite structures that appear in the list have no repetition, up to isomorphism; indeed, they all have distinct cardinalities. □

Lemma 10.2. *The enumeration $(Z_i)_{i\in\omega}$ produced at Phase 2 has the following properties:*

(1) *It contains no repetition, up to isomorphism.*
(2) *It includes all isomorphism types which appear in $(L_i)_{i\in\omega}$.*
(3) *It includes all isomorphism types of infinite junk structures.*
(4) *It satisfies $(b1)$ and $(b2)$ from 9.2.2.*

Proof. (1), (2), and (3) follow from Proposition 9.4, and (4) is an immediate consequences of Modification 1 and the analysis contained in Subsection 8.4; see also 9.2.2.

□

Lemma 10.3. *The enumeration $(B_i)_{i\in\omega}$ produced at Phase 3 has the following properties:*

(1) *It contains no repetition, up to isomorphism.*
(2) *It includes all isomorphism types which appear in $(Z_i)_{i\in\omega}$.*
(3) *It includes all isomorphism types of finite groups.*

Proof. This is a reformulation of Proposition 9.8. □

Combining the three lemmas above, we conclude that $(B_i)_{i\in\omega}$ is a Friedberg enumeration of almost all computable Ulm type $\leq n$ groups. This enumeration does not include the following special isomorphism classes of groups, namely:

(1) abelian p-groups of Ulm type 1 which correspond to equivalence structures having finitely many classes at least one of which is infinite, and
(2) abelian p-groups corresponding to eventually bounded equivalence structures having infinitely many classes.

These two isomorphism classes have a combined uniformly computable Friedberg enumeration which we denoted by $(M_i)_{i\in\omega}$. By merging $(M_i)_{i\in\omega}$ with $(B_i)_{i\in\omega}$ we obtain a computable Friedberg enumeration of all computable Ulm type $\leq n$ abelian p-groups, as desired.

References

[AK00] C. Ash and J. Knight. *Computable structures and the hyperarithmetical hierarchy*, volume 144 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam, 2000.

[AKO] C. Ash, J. Knight, and S. Oates. Recursive abelian p-groups of small length. Unpublished.

[AT11] Vikraman Arvind and Jacobo Torán. Solvable group isomorphism is (almost) in NP $\cap$ conp. *TOCT*, 2(2):4:1–4:22, 2011.

[Bae37] R. Baer. Abelian groups without elements of finite order. *Duke Math. J.*, 3(1):68–122, 1937.

[DKT11] R. Downey, A. Kach, and D. Turetsky. Limitwise monotonic functions and applications. In *Proceedings of STACS 2012*, pages 56–85, 2011.

[DM08] R. Downey and A. Montalbán. The isomorphism problem for torsion-free abelian groups is analytic complete. *J. Algebra*, 320(6):2291–2300, 2008.

[DM14] Rodney Downey and Alexander G. Melnikov. Computable completely decomposable groups. *Trans. Amer. Math. Soc.*, 366(8):4243–4266, 2014.

[DMN16] Rodney Downey, Alexander G. Melnikov, and Keng Meng Ng. Abelian p-groups and the halting problem. *Ann. Pure Appl. Logic*, 167(11):1123–1138, 2016.

[DMN17] Rodney G. Downey, Alexander G. Melnikov, and Keng Meng Ng. A Friedberg enumeration of equivalence structures. *J. Math. Log.*, 17(2):1750008, 28, 2017.

[EG00] Y. Ershov and S. Goncharov. *Constructive models*. Siberian School of Algebra and Logic. Consultants Bureau, New York, 2000.

[Fri58] Richard M. Friedberg. Three theorems on recursive enumeration. I. Decomposition. II. Maximal set. III. Enumeration without duplication. *J. Symb. Logic*, 23:309–316, 1958.

[Fuc70] L. Fuchs. *Infinite abelian groups. Vol. I*. Pure and Applied Mathematics, Vol. 36. Academic Press, New York, 1970.

[GK02] S. Goncharov and J. Knight. Computable structure and antistructure theorems. *Algebra Logika*, 41(6):639–681, 757, 2002.

[His81] N. Hisamiev. Criterion for constructivizability of a direct sum of cyclic p-groups. *Izv. Akad. Nauk Kazakh. SSR Ser. Fiz.-Mat.*, (1):51–55, 86, 1981.

[Kap69] I. Kaplansky. *Infinite abelian groups*. Revised edition. The University of Michigan Press, Ann Arbor, Mich., 1969.

[Khi92] N. Khisamiev. Constructive abelian p-groups. *Siberian Adv. Math.*, 2(2):68–113, 1992.

[Khi98] N. Khisamiev. Constructive abelian groups. In *Handbook of recursive mathematics, Vol. 2*, volume 139 of *Stud. Logic Found. Math.*, pages 1177–1231. North-Holland, Amsterdam, 1998.

[Kie13] Jonathan A. Kiehlmann. Classifications of countably-based abelian profinite groups. *J. Group Theory*, 16(1):141–157, 2013.

[KKM13] I. Kalimullin, B. Khoussainov, and A. Melnikov. Limitwise monotonic sequences and degree spectra of structures. *Proc. Amer. Math. Soc.*, 141(9):3275–3289, 2013.

[KNS97] B. Khoussainov, A. Nies, and R. Shore. Computable models of theories with few models. *Notre Dame J. Formal Logic*, 38(2):165–178, 1997.

[Kur60] A. Kurosh. *The theory of groups*. Chelsea Publishing Co., New York, 1960. Translated from the Russian and edited by K. A. Hirsch. 2nd English ed. 2 volumes.

[LMS18] Karen Lange, Russell Miller, and Rebecca M. Steiner. Classifications of computable structures. *Notre Dame J. Form. Log.*, 59(1):35–59, 2018.

[Mal61] A. Mal′cev. Constructive algebras. I. *Uspehi Mat. Nauk*, 16(3 (99)):3–60, 1961.

[Mel14] Alexander G. Melnikov. Computable abelian groups. *Bull. Symb. Log.*, 20(3):315–356, 2014.

[Mel17] Alexander Melnikov. Computable topological groups and pontryagin duality. page 1, 08 2017.

[MN18] Alexander G. Melnikov and Keng Meng Ng. Computable torsion abelian groups. *Adv. Math.*, 325:864–907, 2018.

[Pon34] L. Pontrjagin. The theory of topological commutative groups. *Ann. of Math. (2)*, 35(2):361–388, 1934.

[Rab60] M. Rabin. Computable algebra, general theory and theory of computable fields. *Trans. Amer. Math. Soc.*, 95:341–360, 1960.

[Rog77] L. Rogers. Ulm's theorem for partially ordered structures related to simply presented abelian p-groups. *Trans. Amer. Math. Soc.*, 227:333–343, 1977.

[Rog87] H. Rogers. *Theory of recursive functions and effective computability*. MIT Press, Cambridge, MA, second edition, 1987.

[Sac90] Gerald E. Sacks. *Higher recursion theory*. Perspectives in Mathematical Logic. Springer-Verlag, Berlin, 1990.

Victoria University of Wellington
Email address: `Rod.Downey@msor.vuw.ac.nz`

Massey University
Email address: `alexander.g.melnikov@gmail.com`

Nanyang Technological University
Email address: `kmng@ntu.edu.sg`