ITERATED EFFECTIVE EMBEDDINGS OF ABELIAN p-GROUPS

RODNEY DOWNEY, ALEXANDER G. MELNIKOV, AND KENG MENG NG

Abstract. The paper contributes to the theory of recursively presented (see Higman [22]) infinitely generated abelian groups with solvable word problem. Mal'cev [35] and independently Rabin [39] initiated the study of such groups in the early 1960's.

In the paper we develop a technique that we call *iterated effective embeddings*. The significance of our new technique is that it extends the iteration technique from the realm of iterated $\mathbf{0}''$ arguments to iterated $\mathbf{0}'''$ ones. This is a new phenomenom in computable algebra. As an illustration, we use this technique to confirm and extend a 30 year old conjecture of Ash, Knight and Oates [2].

More specifically, Ash, Knight and Oates [2] conjectured that there exists a reduced abelian p-group of Ulm type ω such that its effective invariants, limitwise monotonic functions, are not uniform. We construct a computable reduced abelian p-group of Ulm type ω having its invariants, limitwise monotonic functions, not only non-uniform but at the maximal potentially possible level of non-uniformity. The result confirms the conjecture in a strong way, and it also provides us with an explanation of why computable reduced p-groups of Ulm type ω seem hard to classify in general.

We also use p-basic trees and their iterated embeddings to solve a problem posed in [4].

1. Introduction

Following Mal′cev [35] and Rabin [39], we say that a countable group H is *constructive* or *computable* if elements of H can be associated with natural numbers so that the group operation becomes a recursive function on these numbers. The above mentioned numbering of the group is called a *computable presentation* or *constructivistion* of the group. Equivalently, a group has a computable presentation if, and only if, the group admits an effective listing of its generators under which the word problem is solvable.

Mal′cev [35] initiated the systematic study of computable abelian groups. Among other results, Mal′cev characterized computable subgroups of $(\mathbb{Q}, +)$, and also showed that the additive group $\bigoplus_{i\in\omega}\mathbb{Q}$ admits more than one computable presentation, up to computable isomorphism. Computable abelian group theory has developed rapidly along with with other branches of effective algebra. These related branches include effective field theory (see Frölich and Shepherdson [16], Rabin [39], Metakides and Nerode [37]), computable Boolean algebras (Goncharov [19], Remmel [40]) and computable linear orders (Downey [11]). Other closely related subjects are the study of effectively presented vector spaces [7, 8, 36] and the theory of computable ordered groups [12, 20]. For early developments in the field of computable abelian groups, see Nurtazin [38], Smith [42], Lin [34], and Khisamiev [26]. We remark that all groups in this paper will be countable abelian groups.

The modern theory of computable abelian groups is one that not only depends on tools from classical computability theory [43] such as priority arguments, but also heavily uses methods of classical abelian group theory [17, 18, 30] and computable model theory [3, 15], as well as tools specific to the field [32, 10, 1]. Standard references for the theory of computable ablelian groups are [32, 13].

1.1. **The subject of the paper.** The underlying problem of the present paper is one that is fundamental virtually in any area of computable model theory.

Characterize computably presentable members in a given class of strucutres.

Typically, a reasonable answer might entail examining the effective content of a classical theorem giving some kind of invariants. In the theory of p-groups there is Ulm's famous classical characterization of the isomorphism types of such groups. We briefly remind this characterization in the next few lines.

Let p be prime, and A an abelian additive group. A non-zero element g of A has infinite height if for every k the equation $p^k x = g$ has a solution in A. Elements of infinite height generate a sub-group A' of A. Iterating this process[1], we can define $A^{(\alpha)}$ for every ordinal α. The quotients $A_0 = A/A'$ and $A_\alpha = A^{(\alpha)}/A^{(\alpha+1)}$ contain only (non-zero) elements of finite height. Since A is countable, there must be a countable α for which

$$A^{(\alpha)} = A^{(\alpha+1)}.$$

The least such α is called the Ulm type of A and is denoted by $u(A)$. If $A^{(u(A))} = \mathbf{0}$, then A is *reduced*.

One can show that the *Ulm factors* A_α are simply direct sums of finite cyclic groups. Such a direct sum can be fully classified by its Ulm invariant which is the multiset of sizes of its elementary cyclic summands. This classification can be generalized:

Theorem 1.1 (Ulm)**.** *The isomorphism type of a countable reduced (abelian) p-group is completely determined by the isomorphism types of its Ulm factors.*

Another way of stating Ulm's theorem is to look at the sequence of dimensions of the quotients $A_\alpha = A^{(\alpha)}/A^{(\alpha+1)}$ viewed as vector spaces over the $\mathbb{Z}_p$, and these numbers form the *Ulm sequence* for A.

Since the Ulm invariants classify p-groups up to isomorphism, we might hope that we can classify computable p-groups by effectivity conditions on Ulm sequences. Thus, when restricted to the class of countable reduced abelian p-groups, the fundamental problem of characterizing computable members reduces to:

Which Ulm invariants correspond to computably presentable groups?

A related question is the following.

If we fail to answer the question above, are there any reasonable invariants corresponding to computable abelian p-groups?

Given that these are natural questions, one would expect that answers are known. Nonetheless, we will see that the question is harder than one might expect.

[1]For a p-group A, the notation A' can be understood in two significantly different ways (namely, recursion-theoretically and group-theoretically), but it will be always clear form the context what exactly we mean.

1.2. **The effective content of Ulm's theorem.** It is easy to see that the if G is a computable p-group then its Ulm type is $\leq \omega_{CK}^1$ which is the first non-computable ordinal. [34, 32]. It is reasonably easy to see that given any computable ordinal α there exists a computable p-group of Ulm type α whose Ulm invariants are uniformly computable. The trouble is that the Ulm sequene for a computable p-group is often far from computable. For example, it would seem that even computing the elements of finite height in a computable group we need quite a strong oracle $\mathbf{0}''$. Khisamiev and independently Ash, Knight and Oates realized that more than uniform computability is actually required. The formation of sets encoding the invariants have to be constructed in a special *limitwise monotonic* way.

Definition 1.2 (Khisamiev [27])**.** A total function F is *limitwise monotonic* if $F = \lambda x. \sup_y f(x, y)$, where f is computable. A set is limitwise monotonic if it is a range of a limitwise monotonic function.

It is easiest to understand Definition 1.2 via the example of the sizes of cells in a computable equivalence relation E. Suppose that C is the set of sizes of such cells and suppose that in E all such sizes are finite. Then E clearly is the range of a limitwise monotonic set. Conversely, if we have a limitwise monotonic set, then it is easy to construct a computable equivalence relation whose cells have exactly those sizes. We remark that whilst all limitwise monotonic sets are Σ_2^0, the limitwise monotonicity of the required cell sizes imposes a restriction, since there are Σ_2^0 sets that are *not* limitwise monotonic [33].

In the case of p-groups, we will need the definition in relativized form. To wit, replace f by a $\mathbf{0}^{(n)}$-computable function in the definition above and obtain the notion of $\mathbf{0}^{(n)}$-limitwise monotonicity. A set is $\mathbf{0}^{(n)}$-*limitwise monotonic* if it is the range of a $\mathbf{0}^{(n)}$-limitwise monotonic function. For infinite Σ_{n+1}^0 sets, the latter is equivalent to *containing* an infinite range of a $\mathbf{0}^{(n)}$-limitwise monotonic function (see, e.g., [29, 21]). Khisamiev and independently and later Ash, Knight and Oates proved:

Theorem 1.3. [31, 2] Let A be a reduced (abelian) p-group of Ulm type $n < \omega$. Then the following are equivalent:

(1) A has a computable copy;
(2) (a) for every $i < n$, the set
$$S(A_i) = \{(m, k) : \text{ at least } k \text{ summands of } A_i \text{ are of order } p^m\}$$
is Σ_{2i+2}^0, and
(b) for every $i < n$, the set
$$\#A_i = \{m : \mathbb{Z}_{p^m} \text{ is a summand of } A_i\}$$
is $\mathbf{0}^{(2i)}$-limitwise monotonic.

The concept of limitwise monotonicity was new to computability theory. Limitwise monotonic functions have found various applications outside the theory of computable groups [6, 21, 28, 23, 5, 24, 25, 9, 33].

This brings us to the first topic of the present paper. Theorem 1.3 only gives a characterization of the Ulm sequences which can be realized in computable p-groups for *finite* Ulm type $n < \omega$, though as we will see in Proposition 5.2 (and as the authors of those papers noted), the methods allow us to construct a computable p-group of Ulm type ω if we replace (b) above by

(b') for every $i < n$, the set

$$\#A_i = \{m : \mathbb{Z}_{p^m} \text{ is a summand of } A_i\}$$

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

The general question of whether Theorem 1.3 holds for groups of Ulm type $\geq \omega$ has been an open problem for over 30 years.

1.3. **The case of Ulm type ω, and the main result.** If a reduced abelian p-group G is computable, the sets $\#G_i$ have to be $\mathbf{0}^{(2i)}$-limitwise monotonic, uniformly in i or not. One possible approach to generalizing Theorem 1.3 would be to establish that the limitwise functions ranging over $\#A_i$ are always uniformly given for any computable p-group A. This is far from true as we will see.

Counting the number of quantifiers (see Fact 2.8) shows that finding (an index for) a $\mathbf{0}^{(2i)}$-limitwise monotonic function ranging over $\#G_i$ takes at most three extra jumps on top of $\mathbf{0}^{(2i)}$; in fact, the property is $\Pi^0_3(\mathbf{0}^{(2i)})$ uniformly in i. We prove that the upper bound (namely, $\Pi^0_{(2i+3)}$) is sharp:

Theorem 1.4. The exists a computable reduced abelian p-group G of Ulm type ω such that the (indices for) $\mathbf{0}^{(2i)}$-limiwise monotonic functions ranging over $\#G_i$ are not uniformly Δ^0_{2i+3}.

Theorem 1.4 enables us to prove a conjecture of Ash, Knight and Oates [2]:

Corollary 1.5. There exists a computable reduced abelian p-group G of Ulm type ω for which $\#G_i$ are not uniformly $\mathbf{0}^{(2i)}$-limitwise monotonic.

That is, the Ulm sequence can be a far from being uniformly given as it is possible for it to be. We have already observed that it is *necessary* that the sets $\#A_i$ are limitwise monotonic (relative to $\mathbf{0}^{(2i)}$) if A is computable. One technical remark about Theorem 1.4 is the following. It gives evidence that characterizing computable p-groups of Ulm type $\geq \omega$ would have to use an iterated $0'''$-construction since using limitwise monotonicity seems unavoidable in any such proof. Whence, any proof would have to use an iterated $0'''$-guessing procedure. Numerous combinatorial and algebraic issues that would have to be addressed make the task look very difficult if not hopeless (see Conclusion).

The key technical tool of the present paper is Proposition 3.4 which gives a uniform embedding performed on top of a strategy potentially having a Π^0_3-outcome. Some elements of the $0'''$-machinery as well as specific purely algebraic techniques are vital for both construction and its verification. This methodology that we call *iterated embeddings* has already found other applications. In particular, we will use the iterated embeddings to solve a open question on categoricity (to be discussed in Subsection 1.4).

Our proof of Theorem 1.4 detours through an effective version of another classification of the Ulm invariants due to Laurel Rogers. Rogers described how to associate a certain kind of tree with a p-group and conversely. That is, we use the representation of p-groups by p-basic trees [41] (to be defined in the next section). Khisamiev's approach to proving Theorem 1.3 was direct and algebraic. The approach of Ash, Knight and Oates [2] was to effectivize the methodology of Rogers. Ash, Knight and Oates [2] showed that if a group G of type $n < \omega$, or has type $\geq \omega$ and has the uniform properties from the Conjecture, then G is computable if and only if it can be represented as a computable p-basic tree.

The group G from Theorem 1.4 possesses a computable p-basic tree that generates it. Thus, we obtain:

Corollary 1.6. *There exists a computable p-basic tree such that the computable group it generates has no uniform sequence of monotonic functions from the Conjecture.*

This is the first instance where such a separation of the properties of the p-basic trees and those of the limitwise monotonic functions inherent in the groups have separated. We remark that our results also highlight the following question due to Ash, Knight and Oates [2].

Question 1.7. *Can every computable abelian p-group be represented by a computable p-basic tree?*

The paper of Ash, Knight and Oates [2] has never appeared in print[2]. Its methods are important to our work and hence we will take this opportunity to give a brief account of their methods in Section 2.

1.4. **A categoricity question.** Recall that a computable structure is Δ^0_α-categorical if any two computable isomorphic copies of the structure are Δ^0_α-isomorphic [3]. Calvert, Cenzer, Harizanov, and Morozov asked (Problem 5.1 in [4]):

Question 1.8. *Let G be a computable abelian p-group isomorphic to $D \oplus H$, where D is a direct sum of finitely many copies of the Prufer group $\mathbb{Z}_{p^\infty}$, and H is a direct sum of cyclic summands of unbounded orders. Can G be Δ^0_2-categorical?*

Using the technique of iterated embeddings of p-basic trees, we answer the question in negative. In fact, we prove more:

Theorem 1.9. Let G be a computable p-group of finite Ulm type n, such that:
(1.) $G^{(n)} \cong \oplus_{j \leq m} \mathbb{Z}_{p^\infty}$, for some $m \in \omega$;
(2.) orders of cyclic summands in G_{n-1} are not bounded.
Then G is not Δ^0_{2n}-categorical.

In the special case when $n = 1$ we get exactly groups satisfying conditions of Question 1.8. The theorem improves earlier results of Dushenin [14] who used complex full approximation techniques to construct non-Δ^0_{2n}-categorical groups in these classes, for small n.

2. Background and conventions

We assume that the reader has a sufficient background in computability theory [43] and computable model theory [3, 15]. We will be using basic notions of abelian group theory, the standard textbooks are [17, 18, 30], but no solid background in abelian group theory is assumed.

[2]Many thanks to Julia F. Knight who kindly gave us her permission to upload the unpublished manuscript onto the web and assign a public link to the location.

2.1. **p-Basic trees.** In mathematical practice, it is convenient to use tree-like diagrams representing abelian p-groups.

Definition 2.1. [41] A p-basic tree is a set X together with an binary operation $\cdot$ of the sort $\{p^n : n \in \omega \setminus \{0\}\} \times X \to X$ 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 every $g \in X$ and $k, m \in \omega$, and
(3) for each nonzero element x in X, there is a positive integer n such that $p^n \cdot x = 0$.

Given a p-basic tree X we can pass to an abelian p-group $G(X)$. We use $X \setminus \{0\}$ as the set of generators, and put $px = y$ into the collection of relations if $p \cdot x = y$ in X. Every countable reduced abelian p-group is generated by some well-founded p-basic tree [41].

Convention 2.2. We usually identify a p-basic tree T, the corresponding tree with a distinguished root, and the abelian p-group generated by T. The reader should keep in mind that a group typically has more than one p-basic tree generating it.

Recall the notion of ordinal tree rank for a well-founded tree: every leaf has tree rank 0, and the tree rank of any other vertex is the least ordinal greater than the ranks of all its successors. Notice that every element in $G(X)$ can be uniquely represented in the form $\sum_i m_i v_i$, where $v_i \in X$ and $m_i \in \{1, \dots, p-1\}$.

Definition 2.3. Suppose X is a well-founded p-basic tree, and $G(X)$ is the corresponding group. The rank of $\sum_i m_i v_i$, where $v_i \in X$ and $m_i \in \{1, \dots, p-1\}$, is the minimum of tree ranks of the v_i in X.

The definition is independent on the choice of the underlying p-basic tree (follows from Proposition 1 of [41]). We will use the following consequence of Definition 2.3 without explicit reference:

Remark 2.4. The collection of tree-ranks that occur in X is the same as the collection of ranks realized in $G(X)$.

A non-zero element has rank $k \in \omega$ if, and only if, it has height k. Non-zero elements having rank $\geq \omega$ are exactly the elements of infinite height. Thus, we could define the Ulm factors using ranks rather than heights. Furthermore, the Ulm invariants of $G(X)$ can be reconstructed using only tree ranks which appear in X [41].

2.2. **Trees which give rise to isomorphic groups.** All our trees grow downwards. In a tree, a chain is *simple* if each node in the chain has at most one successor.

We will not completely describe the congruence relation $\sim$ on trees defined by the rule $T \sim X$ iff $G(T) \cong G(X)$. A detailed analysis of $\sim$ can be found in [41]. We will be using elementary transformations from one tree to another preserving their $\sim$-class:

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

Example 2.5. For instance, imagine a tree on vertices v_1, v_2, v_3 and v_4 such that v_1 is the root having successor v_2, and v_3, v_4 are the only two children of v_2. This tree corresponds to the abelian group

$$B = \langle v_1, v_2, v_3, v_4 : v_1 = 0, pv_2 = v_1, pv_3 = v_2, pv_4 = v_2 \rangle.$$

Notice that the same group can be represented by another tree: for instance, pick v_1, v_2, v_3 and $v_3 - v_4$ as new generators of the group. Both corresponding p-basic trees represent B.

The procedure described above is called *stripping*. We can iterate this process and obtain a *fully stripped* tree representing the same group. The only restriction is that we have to keep *some* sequence below a node witnessing its tree-rank. For example, a fully stripped tree for a group of Ulm type 1 is simply a collection of simple chains attached to 0. Assuming that every countable p-group has a p-basic tree that generates it [41], we have just proved that every countable reduced p-group of Ulm type 1 is isomorphic to a direct sum of cyclic p-groups [30].

Another very special case of the general framework on p-basic trees is stated in the fact below.

Fact 2.6. Suppose T and X are p-basic trees so that 0 has rank ω in X. There exists a p-basic tree V such that $G(V)_0 \cong G(X)$ and $G(V)' \cong G(T)$.

Proof Sketch. Attach infinitely many finite simple chains to every node in T making ranks of vertices in (the image of) T infinite within V. The lengths of the finite chains should be based on the ranks that occur in X. □

Although classically Fact 2.6 is a triviality, the effective analog of it is not straightforward and is the main technical tool of [2].

Lemma 2.7. [2] Let T be a computable p-basic tree of Ulm type 1 in which 0 has tree-rank ω, and let C be any Π^0_2 subtree of $\omega^{<\omega}$ (C is viewed as a p-basic tree). There exists a computable p-basic tree U expanding C such that $U_0 \cong T$ and $U' = C$.

Theorem 1.3 follows from Lemma 2.7 and the elementary characterization of computable groups isomorphic to sums of cyclic summands (see [32]). The latter involves limitwise monotonicity. Khisamiev claimed an analog of Lemma 2.7 without using p-basic trees and working with groups directly, and the paper of Ash, Knight and Oates never appeared in print. Since we will be using p-basic trees in our construction, and since we will refer to Lemma 2.7, we give an extended sketch of its proof.

Proof. It follows from the characterization of Ulm type 1 groups [32] that there exists a computable limitwise monotonic function f such that

$$range \sup_y f(x, y)$$

is infinite and is contained in the set $\#T_0$ of lengths that occur in T. We also fix a computable predicate R such that $\sigma \in C$ if, and only if, $\exists^\infty y R(y, \sigma)$.

Proof idea. Using f and R, we shall imitate the proof of Fact 2.6. More specifically, if our current approximation to a Π^0_2-predicate $\exists^\infty y R(y, \sigma)$ "fires" on $\sigma \in \omega^{<\omega}$, as well as on all initial segments of σ, by providing new witnesses y for the corresponding strings, we start growing more simple chains below σ using f. The *main*

difficulty is that some of the components we are constructing may become inactive forever, in this case we need to make sure these components do not produce chains of wrong sizes after stripping (recall we need $U_0 \cong T$).

Further assumptions. We list here two extra assumptions that we use to simplify the proof. The assumptions either do not effect the generality or can be removed with some extra work (to be discussed later).

(1) Without loss of generality, we assume $n_x \leq \sup_y f(x,y)$ for a computable increasing sequence $(n_x)_{x\in\omega}$ and also that $\sup_y f(x,y)$ is strictly monotonically increasing as a function of x. The rate of this increasing can be adjusted during the construction, and we can always assume that the next limit is "much larger" than the previous.
(2) For simplicity, we assume that every finite length that occurs in T, up to stripping, actually occurs infinitely often.

Notations used. Without loss of generality, we assume that the copy of $\omega^{<\omega}$ containing C is embedded into another copy of $\omega^{<\omega}$ in which every σ of the original copy has infinitely many clones. The smaller copy of $\omega^{<\omega}$ is denoted by L, and the larger by V.

Recall that we are constructing $U = \bigcup_s U_s$. For every $\tau \in L$ that is ever enumerated into U and which is terminal in U_s, define its *essential length* at stage s, in symbols $[[\tau]]_s$, to be the length that would be represented by τ after the full stripping of U_s. More formally, let $[[\tau]]_s = 0$ if τ is not terminal in U_s, and otherwise let

$$[[\tau]]_s = |\tau| - |\tau_0|,$$

where τ_0 is the longest proper initial segment of τ that has an extension $\tau' \in U_s$ with the property $|\tau'| - |\tau_0| \geq |\tau| - |\tau_0|$. We may define $[[\tau]] = \lim_s [[\tau]]_s$ for every terminal τ of $U = \bigcup_s U_s$.

The requirements. We need to meet, for every $\sigma \in L$, the requirement:

$$R_\sigma : \sigma \in C \to (\exists^\infty \tau \in V \setminus L)\ \sigma \subset \tau$$

together with the global requirement saying that $T_0 \cong U_0$. We would like to split this global requirement into sub-requirements corresponding to different strings of U. Apart from making sure that all sizes from T_0 have been used infinitely many times, we need to meet for each τ the requirement

$$N_\tau : [[\tau]] \textit{ belongs to } \#T_0.$$

We may assume $0 \in T_0$ without any meaningful interpretation.

The strategies. At stage s, we will reserve a sequence $(x_{i,s})_i$ of length at least $card(U_s)$-many large and fresh numbers. We suppress s in $x_{i,s}$. We have $x_i < x_{i+1}$ for every i. The numbers will be put into a stack and used by various R-strategies. We denote x_i corresponding to R_σ by x suppressing its index.

Suppose at stage t we have seen $\exists^{\geq n_{\rho,t}} y R_t(y,\rho)$ for each $\rho \subseteq \sigma$. Here $n_{\rho,t} \in \mathbb{N}$, and $\exists^{\geq n_{\rho,t}} y$ has a clear interpretation. If at stage $s > t$ we see $\exists^{\geq n_{\rho,t}+1} y R_s(y,\rho)$, then we say that R_σ *requires attention* at stage s.

The strategy for R_σ. If at stage s the strategy R_σ requires attention, then enumerate σ into U if σ does not belong to U yet. Then do the following:

i. Enumerate into U a new string $u \in V \setminus L$ disjoint from U_s and extending $\sigma = \emptyset$ such that:

(a) the length of u is $\sup_{y\leq s} f(x,y)$ if $\sigma = \emptyset$, or otherwise
(b) the length of u is $\sup_{y\leq s} f(x,y) - 1$, if $\sigma \neq \emptyset$.

ii. Put the *label* $\boxed{\text{x}}$ onto u.

We are using $\sup_{y\leq s} f(x,y) - 1$ because, as the reader may check on some simple examples, this choice will guarantee $[[u]]_s = \sup_{y\leq s} f(x,y)$. The label $\boxed{\text{x}}$ indicates $[[u]]_s = \sup_{y\leq s} f(x,y)$ and will be used by the N_u-strategy.

The strategy for N_τ. If τ has no label on it, then we do nothing. Otherwise, if τ carries a label $\boxed{\text{x}}$ and $[[\tau]]_s \neq \sup_{y\leq s} f(x,y)$, then:

i. If $[[\tau]]_s \neq \sup_{y\leq s} f(x,y)$ due to $[[\tau]]_s$ increasing (necessarily, by one) since the last stage N_τ was active, then (a.) extend τ by a new $\tau' \in V$ such that $|\tau'| - |\tau| = 1$, and (b.) put the label $\boxed{\text{x}}$ onto τ'.
ii. If $[[\tau]]_s \neq \sup_{y\leq s} f(x,y)$ due to $\sup_{y\leq s} f(x,y)$ increasing by $k > 0$ in value since the last stage N_τ was active, then (a.) extend τ by a new $\tau' \in V$ such that $|\tau'| - |\tau| = k$, and (b.) put the label $\boxed{\text{x}}$ onto τ'.

Construction. At *stage* 0, using V, initiate the enumeration of finite simple chains of lengths $\sup_y f(x,y)$ attached to 0, with infinitely many chains for each x.

Stage s has two substages:

(1) Correct the finite simple chains introduced at stage 0 by extending them within V to longer simple chains if it is necessary, according to $\sup_{y\leq s} f(x,y)$.
(2) Reserve a sequence of large fresh numbers $(x_{i,s})_i$ of length at least $card(U_s)$. Distribute the numbers among $\{R_\sigma : \sigma \in U_s\}$ so that if $\sigma \subset \sigma'$ then σ gets assigned to a number larger than the number associated to σ'. Let the strategies for $(R_\sigma)_{\sigma\in L\restriction s}$ and then $(N_\tau)_{\tau\in V\restriction s}$ act according to their instructions.

Verification scheme. We need to argue that every requirement is met, and that $T \cong U_0$. Every R-requirement is clearly met though, because we act for the sake of σ using longer and longer chains infinitely often only if $\sigma \in C$. Whence, the situation is not much different from what we had in the proof of Fact 2.6.

For the sake of N_τ, we need to show that if τ is terminal in U, then its essential length agrees with the label $\boxed{\text{x}}$ that was put onto τ:

$$[[\tau]] = \sup_y f(x,y).$$

We show that every label $\boxed{\text{x}}$, if ever introduced, can be moved at most finitely often. There are two reasons $\boxed{\text{x}}$ can be moved downwards along the tree.

To see *the first reason*, suppose R_σ was the R-strategy that introduced $\boxed{\text{x}}$ to the construction. Then, if $\sigma \neq \emptyset$, we used a chain u of length $\sup_{y\leq s} f(x,y)-1$ to extend σ. It agreed with N_u because an even longer chain gets adjoined to a predecessor of σ at the same stage (consider some simple examples). If R_σ becomes active again at some later stage s, then an even longer chain is adjoined to σ at stage s. To keep the essential length of the node carrying $\boxed{\text{x}}$ equal to $\sup_{y\leq s} f(x,y)$, we extend u by one extra node, and move the label onto this newly introduced node. We will never need to correct the position of $\boxed{\text{x}}$ again unless $\sup_y f(x,y) > \sup_{y\leq s} f(x,y)$. The latter event is exactly *the second reason* we might correct the position of $\boxed{\text{x}}$.

Thus, after some stage on, the label $\boxed{\text{x}}$ settles on a node v. Using an inductive argument, we can show that in this case $[[v]] = \sup_y f(x,y)$. We can also show that

every string from $U \setminus L$, except for those introduced at stage 0, has an extension carrying a label. This is done by induction as well. Whence, we argue that the actual lengths that occur in U_0 agree with T. Because of the extra assumptions we made about T_0 and f, we get $T = T_0 \cong U_0$. This completes the verification scheme.

Removing the extra assumptions. The extra assumption (1) on the choice of f is removed by a dynamic transformation of f into another (approximation to a) limitwise monotonic f' with $range_x \sup_y f'(x,y) \subseteq range_x \sup_y f(x,y)$. The dynamic transformation can be done during the construction and it has almost no effect on the construction.

The extra assumption (2) can be removed by modifying Stage 0 and Substage (1) of Stage $s > 0$. More specifically, we guess if a certain length is present in T (this is a Σ^0_2-process), and if we think that yes then we introduce this length using a new simple chain extending $\emptyset$. If later we change our guess, we assign a fresh value x to the simple chain and work towards making its length equal to $\sup_y f(x,y)$. We of cause have to be careful and do not use too many simple chains of length $\sup_y f(x,y)$, the available number of chains is determined by T, but this issue can be sorted out dynamically. □

The calculation of the upper bound on the uniformity of limitwise monotonic functions in G is given below:

Fact 2.8. In a computable G of Ulm type ω, the sets $\#G_i$ are uniformly $\Pi^0_3(\mathbf{0}^{(2i)})$-limitwise monotonic. (There exists a $\Pi^0_{(3+2i)}$-sequence indiex sets witnessing limitwise monotonicity.)

Proof of Fact 2.8. Saying that an element $g \neq 0$ has infinite height in G is Π^c_2-statement. We also say that there is no $h \in G$ having infinite height such that $ph = g$. The combined complexity is Π^0_3. Given such an element, we can effectively pass to a limitwise monotonic function with the range $\{n : (\exists a)(h_p(a) = 0 \wedge p^n g = a)\} \subseteq \#G_0$. A relativized version of this argument gives $\Pi^0_3(\mathbf{0}^{(2i)})$ when considering $\#G_i$. □

3. Proof of Theorem 1.4

Our goal it showing that the upper bound given by Fact 2.8 is sharp. Throughout the proof, all groups are reduced abelian p-groups. We will typically identify a p-basic tree and the group it generates, but the reader should keep in mind that non-isomorphic trees may generate isomorphic groups.

The proof is divided into parts. In Subsection 3.1 we formally state the requirements and agree on notations. In Subsection 3.2, we describe how to build G assuming certain uniform operators exist. To define these operators and verify their properties, we need to do quite a bit of preliminary work which is done in the next subsections. In Subsection 3.3 we describe a single Π^0_2-diagonalization strategy in isolation. In Subsection 3.3, the input of the strategy is a simple chain. The situation reflects the simplest way of avoiding a certain length in a p-basic tree in a way that the resulting tree is non-empty and contains elements of finite height. Then, in Subsection 3.4, we modify the basic strategy to a strategy which can handle arbitrary finite tree, not just a simple chain. This modification is required since other strategies may abandon their components, and the diagonalization strategy

will have to deal with these abandoned components which are not necessarily finite simple chains. In Subsection 3.5 we merge the diagonalization strategy and a Π^0_3-guessing procedure. In Subsection 3.6 we construct an auxiliary group F produced by the basic strategy (combined with the Π^0_3-guessing), on a specific input. The construction of G heavily relies on F, since F is used to define the operators (see, e.g., Fact 3.2). The operators are finally defined in Subsection 3.7. The proof is finished in Subsection 3.8.

3.1. **Preliminary analysis, and the requirements.** Recall that for a group G, the set of finite heights which occur in G_i is denoted by $\#G_i$.

We aim to construct a computable reduced abelian p-group G of Ulm type ω for which the indices of functions witnessing limitwise monotonicity of $\#G_i$ are not uniformly $\Delta^0_3(\mathbf{0}^{(2i)})$. Although it is intuitively clear which requirements we need to satisfy, we prefer to formally state them. Let $(F_{0,j})_{j\in\omega}, (F_{1,j})_{j\in\omega}, \ldots$ be the effective listing of all uniformly c.e. sequences of predicates. Based on this listing, and using alternating projections and complementations, we can associate every (uniformly) $\Sigma^0_{(3+2i)}$ sequence of predicates with a single index e, and denote it $(R_{e,i})_{i\in\omega}$, where $R_{e,i}$ is Σ^0_{3+2i} uniformly in i. We say that j witnesses $\mathbf{0}^{(e)}$-limitwise monotonicity of a set X if $f(x) \leftrightharpoons \sup_y \Phi_j(\mathbf{0}^{(e)}; x, y)$ is total, $\mathrm{rng}_x\, f(x)$ is infinite, and

$$\mathrm{rng}_x f(x) \subseteq X.$$

A set S witnesses $\mathbf{0}^{(e)}$-limitwise monotonicity of X if each $j \in S$ witnesses $\mathbf{0}^{(e)}$-limitwise monotonicity of X. We meet, for every e, the requirement:

$$L_e : R_{e,3e} \text{ does not witness } \mathbf{0}^{(6e)}\text{-limitwise monotonicity of } \#G_{3e}.$$

The reason we are using G_{3e} instead of G_e is related to the outcomes of the basic diagonalization strategy and will be explained in the next subsection.

3.2. **Describing G.** We need to construct a computable reduced abelian p-group G of Ulm type ω and meet:

$$L_e : R_{e,3e} \text{ does not witness } \mathbf{0}^{(6e)}\text{-limitwise monotonicity of } \#G_{3e},$$

where the $R_{e,i}$ are Σ^0_{3+2i} uniformly in i.

The strategy for L_e will be discussed in the later subsections. We will relativize this strategy to $\mathbf{0}^{(6e)}$. As a result, L_e will uniformly produce a $\mathbf{0}^{(6e)}$ -computable p-basic tree $A(3e)$ having Ulm type either 1, 2, or 3, depending on the true outcome of L_e.

We wish to construct computable G such that $G_{3e} = A(3e)_0$, for every e. If we succeed, then R_e will be met for every e. We will also make sure $G_{3e+1} \cong G_{3e+2} \cong \bigoplus_{m,n} \mathbb{Z}_{p^m} a_{m,n}$ for every e.

How do we construct G? Instead of constructing the whole G at once, we will construct a uniformly computable sequence of p-basic trees $(B(i))_{i\in\omega}$ such that $B(i)$ is of Ulm type i. We will have $B(i)_{3e} \cong A(3e)$ and $B(i)_{3e-1} = B(i)_{3e-2} \cong \bigoplus_{m,n\in\omega} \mathbb{Z}_{p^m} a_{m,n}$, for every $3e \leq i$. We will set $G = \bigoplus_{i\in\omega} B(i)$.

Why do we homogenize G_{3e+1} and G_{3e+2}? We will need to put the groups $A(j)$ together in a tower (this is the main difficulty), but in some cases our diagonalization modules (L-strategies) will produce "junk" which will contain elements of high rank. We will show that the tree-ranks of the "junk" elements will be less than $\omega \cdot 3$, and they could potentially effect at most two more "levels" of the group. We

circumvent this potential difficulty homogenizing two levels in-between the levels used by different L-strategies.

How do we build $B(i)$? We construct $B(i)$ using operators which map Π^0_2-subtrees of $\omega^{<\omega}$ to computable trees. Recall that every Δ^0_2-tree is isomorphic to a Π^0_1-subtree of $\omega^{<\omega}$, with all possible uniformity.

Given a Π^0_2 p-basic tree D, we can uniformly produce a a computable p-basic tree H such that $H_0 \cong \bigoplus_{m,n} \mathbb{Z}_{p^m} a_{m,n}$ and $G' \cong D$ (see Proposition 3.3). We will also prove that, given the computable tree F constructed by one of the diagonalization strategies (think of L_0 and $A(0)$) and a Π^0_2 subtree C of $\omega^{<\omega}$ such that $C_0 \cong C_1 \cong \bigoplus_{m,n} \mathbb{Z}_{p^m} a_{m,n}$, we can uniformly construct a computable tree U such that $U_0 \cong F_0$ and $U' \cong C$ (see Proposition 3.4). Assuming these operators exist, we can uniformly construct the trees/groups $B(i)$, and then uniformly pass to $G = \bigoplus_{i\in\omega} B(i)$. The group G will have the desired properties.

3.3. **The basic strategy.** In this subsection we describe the basic diagonalization strategy. The strategy will be then modified, and then merged with a Σ^0_3-guessing procedure.

Recall that for a group G, the collection of finite heights which occur in G_0 is denoted by $\#G_0$. Suppose we wish to uniformly construct a computable group G in which $\#G_0$ is infinite *and* not limitwise monotonic via $\sup_y f(x,y)$ for a given (partially) computable f:

$$\#G_0 \neq range\, \lambda x. \sup_y f(x,y).$$

We assume that G initially has no elements of infinite height. We furthermore assume that G is initially (at stage 0) represented by a p-basic tree which is the collection of finite simple chains (i.e., with no splittings) attached to the root.

Note 3.1. We assume that we know the lengths of the chains in advance.

The strategies' main task is to make sure there is no chains of length $\sup_y f(x,y)$ in the tree representing G. In the simplest case, we can work with each simple chain separately, and modify it trying to make sure its length is not equal to $\sup_y f(x,y)$ for some specifically chosen x. Thus, we initially start with a single simple chain attached to 0 of length $k \geq 2$ and having a as its terminal node.

Strategy.

(1) Wait for s and x such that $\sup_{y\leq s} f_s(x,y)$ is defined and equals to k. If we ever see such a computation, introduce a new element h and declare $p^2 h = pa$.
(2) At a stage $s' > s$, let m be largest such that $p^m h = pa$. Wait for a stage $t \geq s'$ such that $\sup_{y\leq t} f_t(x,y) = k + m - 1$. If such a stage is found, introduce a new generator h' and declare $p^{m+1} h' = pa$.

If the strategy proceeds by iteration of (2) above for larger and larger m, we end up with a having tree rank ω. The strategy produces a finite tree, otherwise. In both cases the requirement is clearly met.

3.4. **The modified strategy.** We are aiming to build a group having a computable p-basic tree. The basic strategy described in the previous subsection will not be sufficient for this goal, since other strategies may possibly effect the isomorphism type of G_0 producing "junk". This "junk" will typically be a finite tree attached to the root. Thus, we need explain how the basic strategy deals with an arbitrary finite tree, not simply with a finite simple chain. This situation will occur in Proposition 3.4 which is the key technical tool of the paper.

This, we explain how to meet

$$\#G_0 \neq range\, \lambda x. \sup_y f(x,y),$$

in the case when G_0 is not represented by a collection of finite simple chains. It will be represented, in general, by a disjoint collection of computable finite trees (all sharing the same root 0). Thus, it is sufficient to explain what happens on input a finite p-basic tree which we denote V.

So, at stage 0 we have a finite tree V. The strategy will produce a potentially infinite tree T extending V. Its main task is making sure there are no chains of length $\sup_y f(x,y)$ in T. At stage s, we will have a finite tree $T[s]$, and $T = \bigcup_s T[s]$. The key idea is:

Never add extra chains to elements from $T[s] \setminus V$.

We shall argue that the procedure below implements this idea, and consequently only elements of V will possibly have infinite heights in T. Since V is finite, we will have $T'' = \mathbf{0}$.

Recall that every finite tree can be transformed to an $\sim$-equal collection of finite chains growing from a single root 0. It is possible to effectively trace images of finite chains under such a transformation and see which chains may contribute to the collection of heights realized in $T[s]$ (see, e.g., the definition of $[[\sigma]]_s$ in the proof of Lemma 2.7). We say that $g \in T[s]$ is *dangerous* if it could potentially witness the failure of the requirement, i.e.:

(1) g is a terminal node, and
(2) there exists a finite chain which terminates at g and witnesses that

$$l_{f,s} = \sup_{y \leq s} f_s(x,y)$$

can be realized as a height after the full stripping of $T[s]$.

Modified strategy restricted to $T[s]$. If $g \in T[s]$ is dangerous, then consider the cases:

Case 1. We have $g \in V$. Then add a new element x to $T[s]$ and declare $px = g$.
Case 2. We have $g \in T[s] \setminus V$, and there exists m such that $h = p^{m+1}g$ is in V but $p^m g \in T[s] \setminus V$. Add a new element x to $T[s]$ and declare $p^{m+2}x = h$ (thus also adding $p^k x$ for $0 < k \leq m+1$).

In both cases, declare g not dangerous. Once there are no dangerous elements left, go to the next stage.

Verification. Note that g can not represent height $l_{f,s}$ in T_0. The action adds a chain which either extends g by 1 point (Case 1) or represents the new relation $h = p^{m+1}g$ (Case 2). In the first case g is not an end-point anymore, and can not represent any finite height in $\#T_0$ itself anymore. In the second case, notice that $h = p^{m+1}g$ is a terminal node in V. Thus, in the second case g belongs to a simple

chain of length less than $l_{f,s}$, and will represent a direct summand of order smaller than $l_{f,s}$ (see the preliminary section).

We could argue (possibly modifying the strategy) that the element x can not become dangerous unless l_f changes. But notice that even if x was dangerous without l_f changing, we would repeat the strategy above with x in place of g, using the same h. Eventually we would add a chain of length $l_{f,s}$ below h. Since h is a terminal node in V, that new added chain can no longer represent height $l_{f,s}$. Consequently, its end-vertex can not be declared dangerous unless l_f increases.

If l_f ever stabilizes, we end up with a finite tree T such that $\lim_s l_{f,s} \notin \#T_0$. If l_f keeps increasing forever, we end up constructing a (possibly infinite) tree T containing V such that only elements of the *finite* V can possibly have infinite height. Thus, heights of elements in T' are bounded in T', and consequently $T'' = \mathbf{0}$.

3.5. **The strategy combined with a Π^0_3-guessing.** In this subsection we explain how we diagonalize against a single Σ^0_3 predicate. All procedures in the subsection are effective. In general, we will be relativizing to an appropriate oracle.

Given a Σ^0_3-predicate represented in the form $\{e : \exists x \exists^\infty y U(x, y, e)\}$, where U is c.e., we will guess which pair $\langle e, x\rangle$ is least such that $\exists^\infty y U(x, y, e)$ (if there is any). Each pair $\langle e, x\rangle$ will be associated with a basic diagonalization strategy working with the function having index e. At stage s the basic module associated with $\langle x, e\rangle$ will be working within interval $I_{e,x}[s]$ of size at least $\sup\{n : \exists^n y \leq s\, U_s(x, y, e)\}$ (i.e., the interval is increased if the predicate "fires" again). At stage s we have a partitioning of ω into sub-intervals:

$$I_{0,0}[s], I_{0,1}[s], I_{1.0}[s], \ldots,$$

from left to right, where

$$I_{a,b}[s] = [\, m_{a,b}[s], n_{a,b}[s]\,].$$

We may additionally assume that if $\langle c, d\rangle = \langle a, b\rangle + 1$ then $m_{c,d}[s] \geq n_{a,b}[s] + \langle a, b\rangle + 1$, so that the intervals are sufficiently far apart.

The basic strategy associated with $\langle e, x\rangle$ will also be aiming to introduce its witness, a natural number $l_{e,x,s}$ representing the supremum of $\Phi_e(\cdot, \cdot)$ on some (first found) input (z, w) such that $\Phi_e(z, w) \geq m_{e,x}[s]$:

$$l_{e,x,s} = \inf\{\sup_{z \leq s} \Phi_e(z, w), n_{e,x}[s]\}.$$

We visualize the configuration at a stage s as follows. We have intervals corresponding to the influence of each sub-strategy, and *labels* $l_{e,x,s}$ representing lengths which need to be avoided when constructing a tree. We initialize the basic module associated with $\langle e, x\rangle$ if one of the modules with smaller index increases its interval or newly introduces/grows its label. In this case $m_{e,x}$ will be lifted to a fresh large number (larger than any number seen so far in the construction) and $l_{e,x,s}$ will be set undefined.

Notice each label may move only to a larger value, and every time one of the interval increases in size all larger labels will be removed and then possibly put on numbers which are very large. (It is crucial for the construction.)

3.6. **The definition of F.** The aim of this subsection is proving:

Fact 3.2. There exists a uniform procedure which, given a Σ^0_3 predicate R, produces a computable p-basic tree F of Ulm type at most 2 such that:

a. $F_0 \cong F_0 \oplus F_0$ (thus, every finite height is represented in F_0 by infinitely many elements);
b. R is not a collection of indices of functions witnessing limitwise monotonicity of $\#F_0$.

The rest of the subsection is devoted to the proof of Fact 3.2. The group F is the output of the basic module combined with a Σ^0_3-guessing (see the previous subsection) on input $\bigoplus_{m,n\in\omega} \mathbb{Z}_{p^n} a_{m,n}$ represented by a tree consisting of simple chains growing from the root 0. We additionally assume that the lengths of simple chains in the tree representing $\bigoplus_{m,n\in\omega} \mathbb{Z}_{p^n} a_{m,n}$ are known in advance (Note 3.1).

Before stage s begins we have a finite collection of finite trees $\{V_i[s-1] : i \leq s-1\}$. The tree $V_i[s-1]$ contains either one of the $a_{m,n}$ or is built around a newly introduced simple chain. All the $V_i[s-1]$ share the same root 0.

Construction. At stage s, let each of the sub-strategies indexed by pairs $\langle e, x\rangle \leq s$ act on $V_i[s-1]$, for each $m, n \leq s$, according to the instructions given in Subsection 3.4. One extra restriction is that the basic strategy associated with $\langle e, x\rangle$ is not allowed to use simple chains of sizes $l_{e',x',s}$, for $\langle e', x'\rangle < \langle e, x\rangle$, all other sizes are available[3]. If the label $l_{e,x}$ is removed or is put onto a larger number, we introduce infinitely many simple chains representing this currently unoccupied length and attach them to 0.

The following outcomes are possible:

(e, x, ∞): The interval $I_{e,x}$ grows to infinity with eventually stable left-most point, and the eventually stable witness corresponding to $\langle e, x\rangle$ tends to infinity.

(e, x, k): The interval $I_{e,x}$ grows to infinity with eventually stable left-most point, and the eventually stable witness corresponding to $\langle e, x\rangle$ is stuck at k. It includes the case when eventually no witness can be chosen (an outcome of the form (e, x, f) with symbol f).

g: This Π^0_3-outcome is a global win corresponding to all intervals being eventually finite.

If (e, x, ∞) is the true outcome, no labels to the left of $I_{e,x}$ ever move after a stage s. At every stage $t \geq s$ at which $I_{e,x}$ increases in size, all labels of sub strategies associated with larger pairs will be moved beyond $I_{e,x}$ to fresh large numbers. In fact, they will be lifted up so large that no tree among $V_{m,n}$ which were influenced by their actions will ever be modified by these strategies again. Consequently, we can argue as in Subsection 3.4 and see that in the limit we construct a p-basic tree F with $F'' = 0$.

If (e, x, k) is the true outcome, we will end up with a p-basic tree F such that $\#F_0 = \omega \setminus S$, where S is a finite set containing all eventually stable l-labels. In fact, $F' = 0$ in this case. Similar argument applies when the true outcome is g, but in this case $\#F_0 = \omega \setminus S$ where F is potentially infinite. (Recall that the intervals are sufficiently far apart, thus we do not have the situation when one l-label is an immediate successor of another l-label, say.) In this case we again have $F' = 0$.

[3]Recall that intervals and, hence, labels corresponding to different strategies are sufficiently far apart.

In any of these cases we succeed in constructing a p-basic tree avoiding an index from the given Σ^0_3-set.

3.7. **The operators mapping Π^0_2-trees to computable ones.** Recall that we identify a p-basic tree with the group it generates. We will need the following two propositions. The first proposition is a special case of a known technical result [2, 31] (stated in Lemma 2.7 above), the second proposition is new.

Proposition 3.3. Given a Π^0_2 p-basic tree D, we can uniformly produce a computable p-basic tree H such that $H_0 \cong \bigoplus_{m,n\in\omega} \mathbb{Z}_{p^m} a_{m,n}$ and $H' \cong D$.

Proof. Adjoin more finite chains of greater length below x is there is more evidence that $x \in D$. Since *all* finite lengths may occur, no further work needs to be done. Also immediately adjoin infinitely many chains of each finite size to the root. □

Throughout this subsection, the p-basic tree from Fact 3.2 will be called *R-avoiding* and will be denoted by F.

Proposition 3.4. There exists a uniform procedure which, given a Σ^0_3 predicate R and a Π^0_2 p-basic tree C such that $C_0 \cong C_1 \cong \bigoplus_{m,n\in\omega} \mathbb{Z}_{p^m} a_{m,n}$, produces a computable p-basic tree U such that $U_0 \cong F_0$ and $U' \cong C$. (Here F is R-avoiding.)

Proof. For future convenience, we modify outcomes described in Subsection 3.6 by splitting them further. We have (e, x, ∞_i) indicating that there has been i stages at which some $\langle e', x'\rangle < \langle e, x\rangle$ increased its interval or moved/newly introduced its l-label. Similarly, we have (e, x, ∞_i), with similar interpretation. This modification will allow us to permamentrly abandon certain blocks in the construction.

We construct a computable group U represented by a computable p-basic tree. The group will be of the form

$$U = F \oplus \bigoplus_{\alpha} H(\alpha),$$

where α ranges over all outcomes of the procedure avoiding R, and F is the group (p-basic tree) of Ulm type at most 2 given by Fact 3.2. If α is the true outcome of the procedure then $H(\alpha)' \cong C$, and $H(\alpha)''' = 0$ otherwise. Additionally, $\bigcup_\alpha \#H(\alpha)_0 \subseteq \#F_0$, whence the diagonalization against R will be successful. It is sufficient to uniformly and independently construct the trees $H(\alpha)$ for different α. We explain the simpler case when $\alpha \neq g$ first, and then describe $H(g)$.

The case of $\alpha \neq g$. Notice that all outcomes $\alpha \neq g$ have the property that the whole $H(\alpha)$ is either $\mathbf{0}$ or will be forever abandoned if α is not the true outcome. If $\alpha = (e, n, \infty_i)$ for some i, then it will be using only lengths that are too small compared to $l_{e,x,s}$ a stage s. It will additionally make sure that lengths $l_{e',x',s}$ for $\langle e', x'\rangle < \langle e, x\rangle$ are not present in $\#H(\alpha)[s]$, at every s. Similarly, if $\alpha = (e, n, k_i)$ of $\alpha = (e, n, f_i)$, we make progress in approximating C but not using chains of lengths $l_{e',x',s}$ for $\langle e', x'\rangle \leq \langle e, x\rangle$. The rest is the same as in Proposition 3.3.

If one of the intervals $I_{e',x'}$ ever increase, or one of the $l_{e',x'}$ ever is assigned to a new number, or if the current guess on $l_{e,x}$ was finitary and now changed, then the whole $H(\alpha)$ will be permanently abandoned. We will then follow the modified basic strategy (Subsection 3.5) on the finite tree that α left behind. Recall that a newly active strategy will always be using labels which are too large compared to the tree. A straightforward induction shows that, if α is the true outcome then $H(\alpha)' \cong C$,

and $H(\alpha)'' = \mathbf{0}$ otherwise (iterate the argument in Subsection 3.5 finitely many times). In both cases clearly $\#H(\alpha)_0 \subseteq \#F_0$.

The Π^0_3-outcome g and $H(g)$. The procedure constructing $H(g)$ will be working within a copy of $\omega^{<\omega}$ which may be viewed as a complete ω-branching tree growing downwards, with root $\emptyset$. At the end, the Π^0_2-tree C will be imaged into $\omega^{<\omega}$ so that nodes of depth n in C are mapped to nodes of depth n in $\omega^{<\omega}$(i.e., level-by-level).

At stage s, the procedure believes that all sizes except for the ones in $Y = \{l_{e,x,s} : e, x \leq s\}$ are available, and it will be using chains of lengths not in Y, unless Y changes. It will add finite chains to elements in $\omega^{<\omega}$ currently corresponding to C, following the usual strategy. If an element is indeed in C then it will be put into $H(\alpha)'$ in the limit. Recall that $l_{e,x,s} \leq l_{e,x,t}$ for $t \geq s$.

Construction. At a stage s, only one of the three situations (and corresponding actions) may occur:

(1) *One of the $I_{e,x}$ increases in size.* In this case all sub-strategies associated with $\langle e', x' \rangle > \langle e, x \rangle$ lift their intervals up to large fresh numbers. For every $\sigma \in \omega^{<\omega}$ such that $|\sigma| = \langle e, x \rangle$, permanently abandon all τ's extending σ which have ever been used by the construction. Approximating C will now proceed under σ within the segment of the Baire space extending σ *disjoint* from all such τ's. (Note: The only reason we may again visit τ or its extension is due to $l_{e',x'}$ increasing for some $\langle e', x' \rangle \leq \langle e, x \rangle$, since all other l-labels will be too large.)

(2) *One of the $l_{e,x}$ increases within a stable $I_{e,x}$.* In this case all sub-strategies associated with $\langle e', x' \rangle > \langle e, x \rangle$ lift their intervals up to large fresh numbers. Then we follow the generalized basic strategy (Subsection 3.4) possibly adding further splittings to chains that could potentially represent size $l_{e,x,s}$. We add a finite chain to a predecessor of τ if there exists a scenario in the construction[4] in which τ could potentially represent size $l_{e,x,s}$. These include only finitely many options, since there are only finitely many initial segments of τ. We use only simple chains of sizes $< l_{e,x,s}$ not in $\{l_{e',x',s} : \langle e', x' \rangle < \langle e, x \rangle\}$.

(3) *$I_{e,x}$ and $l_{e,x}$ are stable, for $\langle e, x \rangle \leq s$.* In this case we make progress in approximating C. We use chains of lengths not in $\bigcup_{\langle e,x \rangle} I_{e,x,s}$ and adjoin chains of these sizes to σ if our current guess is $\sigma \in C$. We also ensure that if a new simple chain of length y is added below a node σ, then
   (i.) $y + i \notin \bigcup_{\langle e',x' \rangle} I_{e,x,s}$ for each $i \leq |\sigma|$;
   (ii.) y is larger than the maximal length among the chains already extending σ, if there are any.
   (Note: Recall that we have reserved plenty of sizes in-between the intervals.)

Verification for $H(g)$. There are two cases in which a segment of the tree built by the procedure can be *abandoned* by the construction. The first case is when the Π^0_2-predicate representing C is eventually silent on input a. The finite subtree extending $\sigma \in \omega^{<\omega}$ associated to a may never be visited again for the sake of approximating C. The second case corresponds to one of the $I_{e,x}$ becoming active again; in this case all subtrees built by the strategy and rooted at level $\langle e, x \rangle + 1$ of $\omega^{<\omega}$ will never be active again for the sake of C-approximation.

[4]This depends on which $\rho \subseteq \tau$ truly represent an element of C.

Remark 3.5. Suppose T is a subtree which looks abandoned (first case, Σ^0_2) or is permanently abandoned (second case, Σ^0_1). It could have happened that the finite lengths which occur in the tree, after stripping, were not allowed (equal to l-labels). For instance, in [2, 31] (see Lemma 2.7) one has to extend the longest chain present in T carefully using the limitwise monotonic function, and then monitor the construction. We do not have to do that in our construction because of (i.) and (ii.) in (3), unless one of the sufficiently small l-labels moves to a larger number.

The construction is organized so that there are only two cases at which we might have to act due to $l_{e,x}$ increasing:

Case 1 We permanently left behind a finite tree T due to $I_{e,x}$ or some other interval increasing. Starting from this stage, we follow the generalized basic strategy (Subsection 3.4) in our actions on this finite tree. The tree will never be used to approximate C.

Case 2 We have left behind a finite tree due to C being silent on one of its inputs, say on c. The node σ currently representing c in Baire space may have arbitrary long finite chains attached to it. We follow the basic strategy (Subsection 3.3) on each of the chains, thus possibly further branching some of the chains extending σ. Note that σ may be visited again due to a new C-activity on c.

In Case 1, the are only finitely many l-labels which are small enough and can potentially force us to add new simple chains to T. Based on this idea, we prove:

Claim 3.6. *In Case 1, tree-ranks of nodes in T are bounded by $\omega+k$ at every stage of the construction, for some fixed $k \in \omega$ not depending on the stage (but depending on the tree).*

Proof. Suppose T is abandoned at stage s. There are only finitely many l-labels which can potentially increase the rank of a node $\sigma \in T$. If all of these labels are eventually stable or too large, the rank of $\sigma \in T$ is finite in the limit. Let l_i be the least among these labels which tends to infinity. It follows that all labels l_j with $j > i$ are lifted up to large numbers at a stage $t \geq s$, and they can not effect the ranks of nodes in T anymore. We may suppose t is a stage after which all labels less than l_j are stable. Let k be the maximal length in the tree T', where T' consists of T and all chains added for the sake of avoiding l-labels at stages $\leq t$. From stage t on, we follow the modified diagonalization strategy (Subsection 3.4). Thus, the ranks of nodes in T' are at most ω in the limit. □

Remark 3.7. The same argument shows that a simple chain added by the procedure when approximating C within $\omega^{<\omega}$ (see (3) of the construction) will be expanded to a tree V having $V'' = 0$.

In Case 2, the worst scenario is when σ representing c is of length smaller than the least $\langle e, x\rangle$ for which $I_{e,x}$ grows to infinity. Then C may fire on c in-between $I_{e,x}$-expansionary stages, and longer simple chains will be added to σ. Then these chains will become infinitely branching due to $l_{e,x,s}$ increasing. In this case we end up with σ having rank $\omega \cdot 2 + k$, for some $k \leq \langle e, x\rangle$. We summarize these ideas in the claim below:

Claim 3.8. *Suppose the true outcome of the main R-avoiding procedure is not $\boldsymbol{g}$. Then $H(g)''' = 0$.*

Proof. Suppose the true outcome is of the form $(e, x, \cdot)$. It corresponds to the case when every σ having length $\geq \langle e, x\rangle + 1$ ever introduced by the construction is permanently abandoned at some stage. Then every σ exceeding $\langle e, x\rangle + 1$ in length may have rank at most $\omega + k$, for some k (Claim 3.6 or Remark 3.7). Whence, every node in $\omega^{<\omega} \upharpoonright \langle e, x\rangle + 1$ has successors of ranks bounded by $\omega \cdot 2$. Therefore, the rank of the root $\emptyset$ is at most $\omega \cdot 2 + \langle e, x\rangle + 1$. □

Finally, we prove:

Claim 3.9. *If* $\boldsymbol{g}$ *is the true outcome of the* R*-avoiding procedure, then* $H(g)' \cong C$.

Proof. Observe that the only reason a permanently abandoned node (Case 1) may have an infinite rank is when one of the l-labels tend to infinity, which is not the case. Therefore, all permanently abandoned nodes may contribute only to $H(g)_0$. The same argument applies if a node is abandoned due to C being eventually silent on the corresponding input (Case 2).

We need to verify that, if $c \in C$, ranks of nodes in the simple chains added to a node σ representing c will be kept finite in the limit. Note that the simple chain can be further branched, using chains of smaller length, only due to one of sufficiently small l-labels moving. Only finitely many labels may force us to further branch the simple chain, all other labels will occupy numbers which are too large. All labels, if defined, have to be eventually settled at finite locations. Thus, we may potentially end up with a finite tree properly containing the original simple chain. It follows that σ will have rank at least $\omega + |\sigma|$ if $c \in C$. It may have a larger rank *only if* there exists $c' \in C$ extending c . □

We have verified that $H(g)' \cong C$ if g is the true outcome, and $H(g)''' = 0$, otherwise. It is also clear from the construction that in both cases $\#H(g)_0 \subseteq \#F_0$. It completes the verification for $H(g)$. □

3.8. **Finalizing the proof.** Using Fact 3.2, we can produce a uniformly $\Pi^0_{(6e)}$-sequence $(A(3e))_{e\in\omega}$ of $R_{e,3e}$-avoiding p-basic trees. By Propositions 3.3 and 3.4, there exist a uniformly computable sequence $(B(i))_{i\in\omega}$ of computable p-basic trees such that, for each $k \leq i$, $B(i)_k \cong A(k)$ if $k = 3e$, and $B(i)_k \cong \bigoplus_{m,n} \mathbb{Z}_{p^m} a_{m,n}$ otherwise.

We set G equal to $\bigoplus_{i\in\omega} B(i)$. By the definition of G, and since $A(3e) \cong A(3e) \oplus A(3e)$ for every e, the requirement L_e is met for each e. Since the operation of taking a direct sum (p-basic trees, defined naturally) is uniform, and the p-basic trees $(B(i))_{i\in\omega}$ are computable uniformly in i, the p-basic tree for G is computable.

4. An application of p-basic trees to categoricity

In this section, we use the machinery of p-basic trees to prove:

Theorem 1.9. Let G be a computable p-group of finite Ulm type n, such that:
(1.) $G^{(n)} \cong \oplus_{j\leq m} \mathbb{Z}_{p^\infty}$, for some non-zero $m \in \omega$;
(2.) orders of cyclic summands in G_{n-1} are unbounded.
Then G is not Δ^0_{2n}-categorical.

Proof. The group G can be written as

$$G = H \oplus D,$$

where $D \cong \oplus_{j\leq m}\mathbb{Z}_{p^\infty}$, and H is reduced of type n. We need a lemma which is of some independent interest. Perhaps, the lemma is not new, but we could not find it in the literature.

Lemma 4.1. Suppose H is a computable reduced abelian p-group of finite Ulm type. Then H has a computable copy if, and only if, $G = H \oplus \oplus_{j\leq m}\mathbb{Z}_{p^\infty}$ has a computable copy.

Proof. The right-to-left implication is elementary. We prove the left-to-right implication. Recall that the socle C_H of an abelian p-group H is the $\mathbb{Z}_p$-vector space of its elements of order p. Since $\mathbb{Z}_p$ is a finite field, and since every computable abelian group can be viewed as a computable module over $\mathbb{Z}_p$, we can decide linear dependence in C_H. Recall that $H^{(0)} = H$ and $H^{(k+1)} = (H^{(k)})'$ consists of elements of infinite p-height in $H^{(k+1)}$. Suppose that n is the Ulm type of H; we have $G^{(n)} \cong \oplus_{j\leq m}\mathbb{Z}_{p^\infty}$. We would like to apply Theorem 1.3 to H, for that purpose we need to produce limitwise monotonic functions; we also show that the Ulm invariants of H are at the right levels of the arithmetical hierarchy. The latter is straightforward. To produce limitwise monotonic functions, we use a non-uniform argument.

Consider the $0^{(2n-2)}$-computable group $A = G^{(n-1)} \cong H^{(n-1)} \oplus (\oplus_{j\leq m}\mathbb{Z}_{p^\infty})$. Note that $H^{(n-1)} = H_{n-1}$, so we need isolate this group within A. Non-uniformly, fix the finite subspace $C_G \cap D$ of the socle C_G of G, consisting of the elements of $D = \oplus_{j\leq m}\mathbb{Z}_{p^\infty}$. Effectively in $0^{(2n-2)}$, list elements of $C_A \setminus (C_A \cap D)$. For every element $g \in C_A \setminus (C_A \cap D)$, its p-height is equal to the p-height of its projection onto the reduced component of A. (Although the projection is not unique, the height is clearly independent of the choice of the reduced complement.) Thus, we can define a $0^{(2n-2)}$-limitwise monotonic function approximating p-heights that occur in $C_A \setminus (C_A \cap D)$. These p-heights are the same as in $C_{H^{(n-1)}}$.

To define a $0^{(k)}$-limitwise monotonic function for H_k, where $0 \leq k < (n-1)$, we pick an element g in $G^{(k+1)}$ that is not in $G^{(k+2)}$. The subgroup of G_k generated by $\{h_1 - h_2 : ph_1 = ph_2 = g\}$ is isomorphic to an infinite pure subgroup of H_k (in fact, it is isomorphic to H_k). Therefore, we can define a $0^{(k)}$-limitwise monotonic function for $\{h_1 - h_2 : ph_1 = ph_2 = g\}$; this limitwise monotonic function will have an infinite range within the corresponding Ulm invariant of H.

To finish the proof of the lemma, it remains to apply Theorem 1.3 to H. □

Note that the lemma above provides us with effective invariants for H in the sense of Theorem 1.3.

For the rest of the proof we fix the standard computable presentation of $\omega^{<\omega}$ by a computable collection of finite tuples of ω. The lemma below is well-known.

Lemma 4.2. There exists a computable subtree of $\omega^{<\omega}$ that has a unique path that computes the canonical Π^0_2-complete set $Tot = \{e : \Phi_e \text{ is total}\}$.

Sketch. We explain how we can produce a c.e. subtree of $\omega^{<\omega}$, and then we say how we can produce a computable subtree. We will make sure that m is the least element of Tot if and only if the string $(\langle m, i\rangle)$ is on the infinite path, for some i (here $\langle .,. \rangle$ is the pairing function, and $(m_1, \ldots, m_k)$ is a string with the i'th element m_i). We explain the main strategy below.

Strategy. Whenever Φ_m converges on one more input, initialize all strategies that guess k is the least element of Tot for $k > m$. If the strategy does not already

have a witness, it chooses i least such that $(\langle m, i\rangle)$ has not served it as a witness yet and declares $(\langle m, i\rangle)$ its new witness.

The strategies guessing that (m, n) is the least and the second least elements of Tot respectively, will be working in the subtree rooted in the witness of the strategy that guesses m, etc. The resulting c.e. tree consists of all nodes ever used in the procedure.

To make the tree computable, we encode stages into witnesses, as follows. At the first level of $\omega^{<\omega}$, we will be working with triples $(\langle m, i, s\rangle)$, not with pairs as before. If the strategy guessing whether $m = \mu_x\{x \in Tot\}$ needs to pick a new witness at stage s, it must be of the form $(\langle m, i, s\rangle)$ where i is least never used so far.

It should be clear that the unique path is Π^0_2. It is also clear that the path computes Tot. (Note that the latter is dependent on the specific computable presentation of $\omega^{<\omega}$.) □

It is clear that we can obtain a coding of $\overline{0^{(2n)}}$ into the elements of tree-rank ∞ of an infinitely branching Δ^0_{2n-1}-tree $T \subseteq \omega^{<\omega}$. We need another lemma that can be found in [2].

Lemma 4.3. *Every Δ^0_2-tree is Δ^0_2-isomorphic to a Π^0_1-subtree of $\omega^{<\omega}$.*

Sketch. Whenever the Δ^0_2-tree $T = \lim_s T_s$ changes below a node x, forever abandon the temporary $\omega^{<\omega}$-image of the subtree of T_s rooted in x. □

A straightforward relativization gives that every Δ^0_{2n-1}-tree is Δ^0_{2n-1}-isomorphic to a Π^0_{2n-2}-subtree of $\omega^{<\omega}$. This observation together with the comment preceding the lemma above, we get a Δ^0_{2n-1}-isomorphism taking the Δ^0_{2n-1}-tree T encoding $\overline{0^{(2n)}}$ onto a Π^0_{2n-2}-subtree Γ_{2n-2} of $\omega^{<\omega}$.

Recall that, for each $i < n$, Lemma 4.2 and Theorem 1.3 provide us with a $0^{(2i)}$-limitwise monotonic function and Σ^0_{2i+2}-set that correspond to H_i and thus to G_i. (Recall that $G_n = 0$ since $G^{(n+1)} = G^{(n)}$ by our assumption. Thus we have effective invariants for $G_0, \ldots, G_{n-1}$, the invariants for G_{n-1} are an $0^{(2n-2)}$-limitwise monotonic function and a Σ^0_{2n} set.)

Without loss of generality, we may assume that every node in $\omega^{<\omega}$ has infinitely many successors that do not belong to Γ_{2n-2}. Under this assumption, we can apply Lemma 2.7 using the corresponding $0^{(2n-2)}$-limitwise monotonic function and Σ^0_{2n-2}-set (see Lemma 4.2) and *expand* Γ_{2n-2} to a to a Δ^0_{2n-3}-tree Ξ_{2n-3}. We then use Lemma 4.3 to produce a Δ^0_{2n-3}-isomorphism of Ξ_{2n-3} onto a Π^0_{2n-4}-subtree Γ_{2n-4} of $\omega^{<\omega}$, etc. We end up with a finite chain

$$T \to_{\phi_{2n-1}} \Gamma_{2n-2} \to_{id} \Xi_{2n-3} \to_{\phi_{2n-3}} \Gamma_{2n-4} \to_{id} \ldots \to_{id} \Xi_1,$$

where Ξ_1 is a computable tree, and the injections ϕ_k are Δ^0_k. As a result, T is Δ^0_{2n-1}-isomorphic to a subtree of Ξ_1.

We may view Ξ_1 as a p-basic tree; let U be the corresponding p-group. It is clear that $B = U \oplus (\bigoplus_{j=1,\ldots,(m-1)} \mathbb{Z}_{p^\infty})$ is isomorphic to G. (Indeed, both groups have the same Ulm invariants, and both groups have exactly m Prufer summands $\mathbb{Z}_{p^\infty}$.)

Lemma 4.2 implies that there is another "good" copy A of G where membership to the divisible summand is decidable. Indeed, take a computable copy of the reduced part R of G (Lemma 4.2 implies it exists) and then effectively adjoin m copies of $\mathbb{Z}_{p^\infty}$ to R. To finish the proof of the theorem, it is sufficient to prove:

Claim 4.4. $A \not\cong_{\Delta^0_{2n}} B$.

We prove the claim. Recall that T was "encoding" $\overline{0^{(2n)}}$ into its elements of tree-rank ∞, and B was built "around" T. Towards a contradiction, suppose $A \cong_{\Delta^0_{2n}} B$ via ψ. We produce a $0^{(2n-1)}$-effective procedure of testing whether $i \in \overline{0^{(2n)}}$. It is sufficient to come up with an $0^{(2n-1)}$-effective procedure of checking whether $m \in T$ has tree-rank ∞ (recall we were using Lemma 4.2 to produce T). We argue relative to $0^{(2n-1)}$. Since T is $0^{(2n-2)}$-computable, we can list its elements. Given $m \in T$, use $0^{(2n-2)}$ to find its image g in B. Then m is on the infinite path if and only if g is divisible in B if and only if $\psi(g)$ is divisible in A. Since the latter is decidable, and ψ is $0^{(2n-1)}$, we arrive at a contradiction. □

5. Conclusion

We expect that Theorem 1.4 can be pushed to any computable ordinal. We note that the group constructed in Theorem 1.4 has a complex uniformity property, but we circumvented many algebraic difficulties by specifically choosing Ulm invariants (homogenizing G_{3e+1} and G_{3e+2}) and their representations (Note 3.1, Remark 3.5). It also seems crucial for the construction that the l-labels can only be moved to larger numbers. Dropping at least one of these restrictions would result serious problems such as a simultaneous interaction of infinitely many strategies. It is not surprising that the classification of computable p-groups of Ulm type $\geq \omega$ is a largely unexplored area.

The group witnessing our main result is represented by a computable p-basic tree. We leave open:

Problem 5.1 (Ash, Knight, Oates)**.** Does every computable p-group have a computable p-basic tree representing it?

The answer is "yes" for p-groups of finite Ulm type, and also for some specifically classes of infinite Ulm type p-groups. The authors of the present paper have not agreed on a conjecture for Problem 5.1.

5.1. **A hierarchy of ω-type groups.** Although we have an evidence the general problem of describing computable Ulm type ω groups is hard, in some special cases we may potentially obtain a satisfactory answer. Consider the following hierarchy. Given $m \leq 4$, let $\mathcal{R}^\omega_p(m)$ be the class of computable reduced p-groups A of Ulm type ω in which $\mathbf{0}^{(2n)}$-indices for limitwise monotonic functions ranging over $\#A_n$ are uniformly Δ^0_{2n+m}. It is readily checked that

$$\mathcal{R}^\omega_p(1) = \mathcal{R}^\omega_p(2),$$

methods developed in our paper can be applied to show

$$\mathcal{R}^\omega_p(2) \subsetneq \mathcal{R}^\omega_p(3),$$

and Theorem 1.4 implies

$$\mathcal{R}^\omega_p(3) \subsetneq \mathcal{R}^\omega_p(4).$$

The following result is well-known:

Proposition 5.2. [2] Groups in $\mathcal{R}^\omega_p(1)$ are in $1-1$ correspondence with uniformly $\mathbf{0}^{(2n)}$-limitwise monotonic collections of Ulm invariants.

Proof idea. We can split each limitwise monotonic set into infinitely many infinite disjoint (unitormly) limitwise monotonic subsets. We then uniformly run the proof of Theorem 1.3 and obtain a uniform sequence of computable groups $(H_n)_{n\in\omega}$, using more of the limitwise monotonic disjoint subsets for larger n, and so that all disjoint subsets are used in one of the H_n. We then pass to $\bigoplus_{n\in\omega} H_n$ which has the desired invariants. □

Problem 5.3. Prove an analog of Proposition 5.2 for $\mathcal{R}_p^\omega(2)$.

The first case to consider would be groups G having $\#G_i$ co-finite, or even $\#G_i$ either ω or a co-singleton. There are some obstacles even in this simplest case.

References

[1] B. Andresen, A. Kach, A. Melnikov, and D. Solomon. Jump degrees of torsion-free abelian groups. To appear in Journal of Symbolic Logic.

[2] C. Ash, J. F. Knight, and S. Oates. Recursive abelian p-groups of small length. Unpublished. An annotated manuscript: https://dl.dropbox.com/u/4752353/Homepage/AKO.pdf.

[3] C. J. 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.

[4] Wesley Calvert, Douglas Cenzer, Valentina S. Harizanov, and Andrei Morozov. Effective categoricity of abelian p-groups. *Ann. Pure Appl. Logic*, 159(1-2):187–197, 2009.

[5] Richard J. Coles, Rod Downey, and Bakhadyr Khoussainov. On initial segments of computable linear orders. *Order*, 14(2):107–124, 1997/98.

[6] Barbara F. Csima, Denis R. Hirschfeldt, Julia F. Knight, and Robert I. Soare. Bounding prime models. *J. Symbolic Logic*, 69(4):1117–1142, 2004.

[7] J. C. E. Dekker. Countable vector spaces with recursive operations. I. *J. Symbolic Logic*, 34:363–387, 1969.

[8] J. C. E. Dekker. Countable vector spaces with recursive operations. II. *J. Symbolic Logic*, 36:477–493, 1971.

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

[10] R. Downey and A. G. Melnikov. Effectively categorical abelian groups. To appear in Journal of Algebra.

[11] R. G. Downey. Computability theory and linear orderings. In *Handbook of recursive mathematics, Vol. 2*, volume 139 of *Stud. Logic Found. Math.*, pages 823–976. North-Holland, Amsterdam, 1998.

[12] R. G. Downey and Stuart A. Kurtz. Recursion theory and ordered groups. *Ann. Pure Appl. Logic*, 32(2):137–151, 1986.

[13] Rodney G. Downey. On presentations of algebraic structures. In *Complexity, logic, and recursion theory*, volume 187 of *Lecture Notes in Pure and Appl. Math.*, pages 157–205. Dekker, New York, 1997.

[14] D. Dushenin. Degrees of autostability of type 2 abelian p-groups with divisible component of a finite dimension. To appear in Algebra i Logika (in Russian).

[15] Yuri L. Ershov and Sergei S. Goncharov. *Constructive models*. Siberian School of Algebra and Logic. Consultants Bureau, New York, 2000.

[16] A. Fröhlich and J. C. Shepherdson. Effective procedures in field theory. *Philos. Trans. Roy. Soc. London. Ser. A.*, 248:407–432, 1956.

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

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

[19] Sergei S. Goncharov. *Countable Boolean algebras and decidability*. Siberian School of Algebra and Logic. Consultants Bureau, New York, 1997.

[20] Sergey S. Goncharov, Steffen Lempp, and Reed Solomon. The computable dimension of ordered abelian groups. *Adv. Math.*, 175(1):102–143, 2003.

[21] Kenneth Harris. η-representation of sets and degrees. *J. Symbolic Logic*, 73(4):1097–1121, 2008.
[22] G. Higman. Subgroups of finitely presented groups. *Proc. Roy. Soc. Ser. A*, 262:455–475, 1961.
[23] Denis Hirschfeldt, Russell Miller, and Sergei Podzorov. Order-computable sets. *Notre Dame J. Formal Logic*, 48(3):317–347, 2007.
[24] Denis R. Hirschfeldt. Prime models of theories of computable linear orderings. *Proc. Amer. Math. Soc.*, 129(10):3079–3083 (electronic), 2001.
[25] Denis R. Hirschfeldt, Bakhadyr Khoussainov, and Pavel Semukhin. An uncountably categorical theory whose only computably presentable model is saturated. *Notre Dame J. Formal Logic*, 47(1):63–71 (electronic), 2006.
[26] N. G. Hisamiev. The periodic part of a strongly constructivizable abelian group. In *Theoretical and applied problems in mathematics and mechanics (Russian)*, pages 299–303, 318. "Nauka" Kazakh. SSR, Alma, 1977.
[27] N. G. 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.
[28] Asher M. Kach. Computable shuffle sums of ordinals. *Arch. Math. Logic*, 47(3):211–219, 2008.
[29] I. Kalimullin, B. Khoussainov, and A. Melnikov. Limitwise monotonic sequences and degree spectra of structures. To appear.
[30] Irving Kaplansky. *Infinite abelian groups*. Revised edition. The University of Michigan Press, Ann Arbor, Mich., 1969.
[31] N. G. Khisamiev. Constructive abelian p-groups. *Siberian Adv. Math.*, 2(2):68–113, 1992. Siberian Advances in Mathematics.
[32] N. G. 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.
[33] Bakhadyr Khoussainov, Andre Nies, and Richard A. Shore. Computable models of theories with few models. *Notre Dame J. Formal Logic*, 38(2):165–178, 1997.
[34] Charlotte Lin. Recursively presented abelian groups: effective p-group theory. I. *J. Symbolic Logic*, 46(3):617–624, 1981.
[35] A. I. Mal′cev. On recursive Abelian groups. *Dokl. Akad. Nauk SSSR*, 146:1009–1012, 1962.
[36] G. Metakides and A. Nerode. Recursively enumerable vector spaces. *Ann. Math. Logic*, 11(2):147–171, 1977.
[37] G. Metakides and A. Nerode. Effective content of field theory. *Ann. Math. Logic*, 17(3):289–320, 1979.
[38] A.T. Nurtazin. *Computable classes and algebraic criteria of autostability*. Summary of Scientific Schools, Math. Inst. SB USSRAS, Novosibirsk, 1974.
[39] Michael O. Rabin. Computable algebra, general theory and theory of computable fields. *Trans. Amer. Math. Soc.*, 95:341–360, 1960.
[40] J. B. Remmel. Recursive Boolean algebras. In *Handbook of Boolean algebras, Vol. 3*, pages 1097–1165. North-Holland, Amsterdam, 1989.
[41] Laurel A. Rogers. Ulm's theorem for partially ordered structures related to simply presented abelian p-groups. *Trans. Amer. Math. Soc.*, 227:333–343, 1977.
[42] Rick L. Smith. Two theorems on autostability in p-groups. In *Logic Year 1979–80 (Proc. Seminars and Conf. Math. Logic, Univ. Connecticut, Storrs, Conn., 1979/80)*, volume 859 of *Lecture Notes in Math.*, pages 302–311. Springer, Berlin, 1981.
[43] Robert I. Soare. *Recursively enumerable sets and degrees*. Perspectives in Mathematical Logic. Springer-Verlag, Berlin, 1987. A study of computable functions and computably generated sets.