DEGREE SPECTRA OF UNARY RELATIONS ON $\langle \omega, \leq \rangle$

ROD DOWNEY, BAKHADYR KHOUSSAINOV, JOSEPH S. MILLER, AND LIANG YU

Abstract. A computable presentation of the linearly ordered set $(\omega, \leq)$, where ω is the set of natural numbers and $\leq$ is the natural order on ω, is any linearly ordered set $\mathbf{L} = (\omega, \leq_L)$ isomorphic to $(\omega, \leq)$ such that $\leq_L$ is a computable relation. Let X be subset of ω and X_L be the image of X in the linear order $\mathbf{L}$ under the isomorphism between $(\omega, \leq)$ and $\mathbf{L}$. The degree spectrum of X is the set of all Turing degrees of X_L as one runs over all computable presentations of $(\omega, \leq)$. In this paper we study the degree spectra of subsets of ω.

1. Introduction

Our interest in this paper falls into part of a long term program in computable model theory where we study the *spectrum* problem for relations on various classes of models. Recall that a **computable structure** is one whose domain and basic relations and predicates are uniformly computable. In particular, if the language (equivalently signature) of the structure is finite then computability of the structure is simply equivalent to saying that its domain and all of its basic relations and operations are computable. If $\mathcal{A}$ is a computable structure isomorphic to $\mathcal{B}$ then we say that $\mathcal{B}$ is **computably presentable** and $\mathcal{A}$ is a **computable presentation (or copy)** of $\mathcal{B}$. Note that computable presentations of a given structure, by definition, are isomorphic structures but can exhibit different computability-theoretic behavior. To capture this behavior the concept of the spectrum of a relation has been introduced that we now explain.

Suppose that R is a relation on a computable structure $\mathcal{A}$. We postulate that R is closed under automorphisms of the structure $\mathcal{A}$, that is, $g(R) = R$ for all automorphism g of $\mathcal{A}$. If $\mathcal{B}$ is a computable structure isomorphic to $\mathcal{A}$, then we will let $R_{\mathcal{B}}$ denote the image of R in $\mathcal{B}$. The **spectrum of** R then is defined as the set of all Turing (or perhaps other) degrees of $R_{\mathcal{B}}$ as we run over all computable $\mathcal{B}$ isomorphic to $\mathcal{A}$. The spectrum of a relation describes the algorithmic properties of the relation in different computable presentations of the structure. The assumption that R is closed under automorphisms is an important one. Indeed, think of R as a unary relation that singles out integers in the linearly ordered set of rational numbers $(Q, \leq)$. This ordering

1991 *Mathematics Subject Classification.* Primary 03D45.

Yu was supported by postdoctoral fellowship from the New Zealand Institute for Mathematics and its Applications, NSF of China 10701041. Miller was supported by a Postdoctoral Fellowship from the Marsden Fund. Downey and Khoussainov are supported by Grants from the Marsden Fund. Additionally, this research was begun at the NZIMA meeting in Nelson, January 2004, part of NZIMA CoRE program in Logic and Computation.

has exactly one computable isomorphism type but the spectrum of R consists of all possible Turing degrees.

Our understanding of possible degree spectra for distinguished relations on various structures has had significant advances in recent years. We know that for many relations the typical spectra we would expect to see would consist of a single element or infinitely many. (See e.g. Harizanov [5, 6, 7], Hirschfeldt [8], Goncharov [10], or Moses [14]) We have also made great strides in the construction of models such as graphs, partial orderings, rings, groups, integral domains, etc with many pathological spectra such as 2 element spectra. (See e.g. Hirschfeldt [8], Hirschfeldt, Khoussainov, Shore and Slinko [11].)

In-spite of the advances mentioned in the study of degree spectra of relations, our understanding of the spectra is much more limited for *particular* structures and *particular* relations. For instance, whilst we know that there are computable linear orderings L where the adjacency relation, $Adj(L) = \{(x, y) : x \text{ is adjacent to } y\}$, has either an infinite degree spectrum, or a spectrum consisting of a single element from $\{\mathbf{0}, \mathbf{0}'\}$, (Downey and Moses [4]) or a principal filter computably enumerable Turing degrees (Downey [2]), or every computably enumerable degree *except* $\mathbf{0}$, little else is known. Recall that $\mathbf{0}$ denoted the computable Turing degree. It has recently been shown that the complete degree $\mathbf{0}'$ is always a member of the spectrum of $Adj(L)$ if the spectrum is nontrivial (Downey, Lempp, and Wu [3]).

In this paper instead of investigating the spectra of specific relations (such as $Adj(L)$ in linear orders), we will study relations on the simplest ordering one could imagine. Namely, we will investigate unary relations X on the standard ordering of the natural numbers $(\omega, \leq)$. Note that the standard natural ordering of the order of natural numbers is characterized by the fact that the adjacency relation is computable. Indeed, any two computable presentations of the order of natural numbers in which the adjacency relations are computable must be computably isomorphic. The ordering $(\omega, \leq)$ is well understood from a computability-theoretic point of view. For example, the ordering is Δ^0_2 stable in the sense that if $(L, \leq_L)$ is a computable ordering isomorphic to $(\omega, \leq)$ then the isomorphism between $(\omega, \leq)$ and is $(L, \leq_L)$ in a Δ^0_2-set.

Let X be a unary relation on $(\omega, \leq)$. If $(L, \leq_L)$ is a computable presentation of $(\omega, \leq)$ then the image of X in $(L, \leq_L)$ can be thought simply as coding of X in $(L, \leq_L)$. We denote this image as X_L. The reader might therefore think very little can really happen. Our results indicate that even in such a simple situation, and even for X that have simple descriptions in computability-theoretic terms, we find interesting and unexpected behaviour. We single out this notation:

If $(L, \leq_L)$ is a computable presentation of $(\omega, \leq)$
then the image of X in $(L, \leq_L)$ is denoted by X_L.

Our paper starts by studying the situation where the unary relation X in $(\omega, \leq)$ is computably enumerable (c.e. for short). One of our results show that the degree spectra of c.e. unary relations in $(\omega, \leq)$ is closed upwards in the Turing degrees. This informally tells us that coding is always possible:

Theorem 1.1. *For any infinite and coinfinite c.e. sets X and Y if $X \leq_T Y$, then there is a computable presentation $\mathbf{L}$ of $(\omega, \leq)$ so that $Y \equiv_T X_L$. That is, the degree spectra of c.e. unary relations represented in a standard copy of $(\omega, \leq)$ are all closed upwards in the Turing degrees.*

As an immediate corollary we have the following.

Corollary 1.2. *The degree spectrum of every infinite and co-infinite c.e. set X possesses a Turing complete degree.*

A natural question arises as to whether the spectrum of a c.e. unary relation X contains *all* c.e. Turing degrees. To prove this, it would suffice to show that the image of X in some computable presentation of $(\omega, \leq)$ is computable. It turns out that the situation here is rather surprising as stated in the following result. For the result we mention that a degree $\mathbf{a}$ is **low** if $\mathbf{a}' = \mathbf{0}'$ wheer

Theorem 1.3. *Let X be a non-computable c.e. set. The degree spectrum of X possesses a low c.e. degree. In fact, there exists a computable presentation $\mathbf{L}$ such that X_L is a c.e. set, $X_L \leq_T X$, and X_L is low.*

Lowness of a degree does not guarantee that the degree is computable. Hence, one would like to know a sufficient condition at which the degree spectrum of a relation contains the computable degree. This is answered in the next theorem. For the theorem we recall a c.e. degree $\mathbf{a}$ is called *high* if $\mathbf{a}' = \mathbf{0}''$.

Theorem 1.4. *If X is a c.e. set whose Turing degree is not high, then the degree spectrum of X contains the computable degree. In other words, there exists a computable presentation $(L, \leq_L)$ of $(\omega, \leq)$ in which the copy of X is a computable set.*

We note that Hirschfeldt, R. Miller, and Podzorov independently proved a version of this theorem for low_2 sets X [9].

It turns out that high degrees are source of examples of c.e. sets X whose degree spectra do *not* have the computable degree. The result below converses Theorem 1.4 as follows:

Theorem 1.5. *Every high c.e. degree contains a c.e. set X such that the degree spectrum of X does not contain the computable degree.*

In some sense the latter theorem is the best we could hope for degrees since we can easily establish the following.

Theorem 1.6. *Every c.e. m-degree contains a set X such that the degree spectrum of X contains a computable set.*

Our remaining results are concerned with spectra of general X. It is not too difficult to construct an X whose degree spectrum contains $\mathbf{0}''$ only. This result can be proved by using the Δ^0_2-stability of $(\omega, \leq)$, and a crude coding of the set $\mathbf{0}''$. However, the situation for Δ^0_2-sets X i s much more subtle. Here is one result:

Theorem 1.7. *There is a Δ^0_2 set X such that the degree spectrum of X is $\{\mathbf{0}'\}$.*

This theorem is just a special case of the following more general result. The result is of interest it says that the Turing degree of any set that computes the Halting problem can be realized as the degree spectrum of some unary relation X. Formally:

Theorem 1.8. *Suppose that for a set $B \subset \omega$ we have $\emptyset' \leq_T B$. Then there is a set X whose degree spectrum is exactly $\{deg(B)\}$.*

The next theorem, in certain respect, refines Theorem 1.3 for a particular set and particular Turing degrees:

Theorem 1.9. *There is a Δ^0_2-set X such that images of X in all computable presentations of $(\omega, \leq)$ have low but non-computable degrees.*

Our last result in this paper shows that for every infinite and co-infinite subset X of ω, the degree spectrum of X can not be bounded by the complete Turing degree. An interesting note here is that the unboundedness (by the complete degree) of the degree spectrum is witnessed by two computable linearly ordered sets for *all* infinite and co-infinite subset X. Formally, here is the statement:

Theorem 1.10. *There exist two computable presentations $\mathbf{L_1}$ and $\mathbf{L_2}$ of the linear order $(\omega, \leq)$ such that for all infinite and coinfinite X, we have $\emptyset' \leq_T X \oplus X_{L_1} \oplus X_{L_2}$.*

2. Notation and Terminology

In this section we introduce a set of notations that will be used throughout the paper. Assume that $\mathbf{L} = (\omega, \leq_L)$ is a computable presentations of the order of natural numbers $(\omega, \leq)$.

Definition 2.1. *For the linear order $\mathbf{L} = (\omega, \leq_L)$ and for every x in the domain of $\mathbf{L}$ define:*

- *$Suc(L)(x) = y$ iff $y >_L x$ and for all y' if $y' >_L x$ then $y' \geq_L y$.*
- *$Pre(L)(x) = y$ iff $y <_L x$ and for all y' if $y' <_L x$ then $y' \leq_L y$.*
- *$Ord(L)(x)$ is, by definition, the number of elements y such that $y <_L x$.*
- *Given a number $z \in \omega$, we define the isomorphic copy of z in $\mathbf{L}$ to be z_L. Clearly, $Ord(L)(z_L) = z$.*
- *Given a set $X \subseteq \omega$, the isomorphic copy of X in $\mathbf{L}$ is denoted by X_L. It is clear again that $X_L = \{x_L | x \in X\}$.*

Our other notation is standard and follows Soare [15]. For example $\leq_T$ denotes the Turing reduction. For a basic referenceon the theory of computable linearly ordered sets we refer the reader to Downey [2]. We now start proving theorems stated in the introduction.

3. The Proof of Theorem 1.3

3.1. **Basic strategies.** Given a noncomputable c.e. set X, we will construct a computable presentation $\mathbf{L}$ of $(\omega, \leq)$ such that X_L is a low c.e. set and $X_L <_T X$. Naturally, we will build $\mathbf{L}$ in stages, keeping track of X_s. At each

stage s we will add one or more elements to X_L that is being built. The overall requirements we must satisfy are the following:

$$Q \ : \ X_L \text{ is c.e.}$$
$$N_e \ : \ X \neq \Phi_e^{X_L}$$
$$M \ : \ X_L \text{ is low}$$
$$P \ : \ X_L \leq_T X.$$

We will break down these tasks into more manageable sub-requirements that are explained below.

Meeting requirement Q. Suppose that there exists an element x such that $X(x)[s+1] \neq X_L(x_L)[s+1]$. Then there are two cases:

(1) $X(x)[s+1] = 0$. This case can occur through our efforts to meet other obligations. Since we want X_L to be c.e., in this case we insert a new number between $Pre(L)(x_L)$ and x_L.
(2) $X(x)[s+1] = 1$. In this case we either insert a new number between $Pre(L)(x_L)$ and x_L or put x_L into X_L.

We say an element x is moved at stage $s+1$ if $Ord(L)(x)[s+1] \neq Ord(L)(x)[s]$. Now it will suffice to prove that every element $i \in \omega$ has a final position in $\mathbf{L}$ (i.e. moved only finitely often). Thus, we split the Q-requirement to infinitely many subrequirements:

$$Q_i \ : \ i \in L \text{ will stop moving eventually.}$$

Meeting requirement N_e. To meet N_e we use classical Sacks technique of preserving agreements. To satisfy N_e, we will monitor the length of agreement $\Phi_e^{X_L} = X[s]$. Thus, at stage s, we define:

$$\hat{\ell}_e[s] = \min\{n | X(n)[s] \neq \Phi_e^{X_L}(n)[s]\},$$
$$r_e[s] = \sum_{m \leq \hat{\ell}_e[s]} \phi_e^{X_L}(m)[s],$$
$$R_e[s] = \sum_{e' \leq e} r_e[s].$$

At every stage s, we try to preserve $\hat{\ell}_{e,s}$ by setting up the restraint $R_e[s]$. That is, once we see the length of agreement exceed some value y at some stage s, we will preserve X_L on the use of this computation. In the usual way, we argue that $\ell(e, s) \not\to \infty$, lets X be computable, which it is not.

Of course, the trouble is when a small element say x entered X (whose current corresponding image $x_L[s]$ is below the use of some e-computation to be preserved) at stage $s+1$. Now we have agreed by the restraint, we can not put the element $x_L[s]$ into X_L since $x_L[s]$ less than the restraint. However, x has entered $X[s]$, and hence it needs an image in L. The idea is to shift the isomorphism. We replace $x_L[s]$ with a big number z. That is, we move $x_L[s]$ by defining $x_L[s+1] = z$ and $(x+i)_L[s+1] = (x+i+1)_L[s]$. Were x_L moved infinitely often, then we would fail to satisfy Q since we are destroying a requirement Q_j for some j. A crucial point is if X is noncomputable then it must be infinite and coinfinite. We will argue that this allows us to ensure that

every number will stop moving. The reader should note that the introduction of the new element z as the new image for the number entering X could consequentially cause us to need to add a lot of new elements into L to allow us to rearrange the isomorphism whilst preserving the N_e computations, with a sort of "cascade" effect.

Meeting requirement M. Our argument is finite injury and so via the N_e we will be able to argue *post hoc* that X_L is automatically low.

Meeting requirement P. Once we prove that $\mathbf{L}$ is isomorphic to $(\omega, \leq)$, we can effectively in X decide X_L. To decide x in X_L or not, we just need search a stage s at which $X \upharpoonright (Ord(L)(x)[s]+1) = X[s] \upharpoonright (Ord(L)(x)[s]+1)$. Since $\mathbf{L}$ is isomorphic to $(\omega, \leq)$, there must exist such a stage. After this stage, x will never be moved. So $X_L \leq_T X$.

There will be conflicts between satisfying requirements Q and N. To give more intuition, we provide an idea of meeting and satisfying one requirement N_e.

3.2. **One N requirement.** We will concentrate on a single N requirement, say N_0. During the construction, we use the concept of a *crucial points sequence* (c.p.s.) $\{z_i\}_{i\leq m, m\leq\omega}$.

Definition 3.1. *A crucial points sequence (c.p.s.) is a (finite or infinite) sequence $\{z_i\}_{i\leq m, m\leq\omega}$ which has the following properties:*

(1) $\{z_i\}_{i\leq\omega}$ is a computable set.
(2) $\forall i \leq m(z_i <_L z_{i+1})$.
(3) $\forall z \in \omega \exists i \leq m(z \leq_L z_i)$.
(4) $\forall i \leq m \forall z \leq z_i(z \leq_L z_i)$.

Item (3) assumes that the sequence is an infinite one.

Clearly, if there is some number z moved infinitely often, then there must be a number z' in the c.p.s. moved infinitely often (just pick up a number $z' > z$ in the c.p.s.). We denote the set $\{z | z_i <_L z \leq_L z_{i+1}\}$ by $(z_i, z_{i+1}]_{\leq_L}$.

Choose a stage s after which N_0 does not receive attention. Then $\forall t \geq s(R(0,t) = R(0,s) = R)$. Suppose $R > 0$.

Suppose there is a number moved infinitely often for the sake of N_0. Then there must be a least number y moved infinitely often. The point is that $y \leq R$ since the only numbers that N_0 moves are below the restraint. For more details, see Lemma 3.4. So there must a z_i in the c.p.s. so that $y \in (z_i, z_{i+1}]_{\leq_L}$.

Define $Mov = \{z | z \text{ moved infinitely often}\}$ and $M(y) = Mov \cap (z_i, z_{i+1}]_{\leq_L}$. Then $\forall z \in Mov(z > z_i)$. Select a number $y' \in M(y)$ so that for all $z \in M(y)$, $y' \leq_L z$. Then y' is the least number $\leq R$ moved infinitely often. We will show that this does not happen. Select a stage $t \geq s$ so that for all numbers z with $z <_L y'$ and $z < R$ are not moved after stage t. Then at any stage $t' > t$, the only action to move y' is inserting a big number between y' and $Pre(L)(y')[t']$. Using the fact the X is infinite, we will argue that we will not move y' infinitely often by Subsection 3.1 Case (1) (that is because of new numbers entering X causing y''s movement.). Without loss of generality, at every stage $t' > t$, for every $z = Ord(L)(y')[t']$, $z \in X$. So X includes the

set $\{z|z > Ord(L)(y')[t]\}$ and so is a cofinite set. This contradicts with the assumption that X is noncomputable and so coinfinite. Therefore, we will stop moving y' eventually.

We will see that $X_L \leq_T X$ since we can decide whether any number in the c.p.s. will be moved with the help of X.

3.3. **The Construction.** We only distribute the priority ordering to N requirements by setting $N_e < N_{e+1}$ for all e. For Q, there is a dynamic priority ordering during the construction. At every stage s, we will construct a linear order $\mathbf{L}[s]$, a finite c.p.s. set $CPS[s] = \{z_i\}$ and restraint function $r_e[s]$. Eventually, $CPS = \cup_s CPS[s]$.

At stage -1, define the c.p.s. set $CPS[-1] = \{0\}$ and $X_L[-1] = \emptyset$. Define $r_e[-1] = -1$ and $\hat{\ell}_e[-1] = -1$, for every $e \geq 0$.

At stage $s+1$, we define a block set by induction on e.

- $BL_0[s+1] = \{z_i|z_i \in CPS[s] \text{ and } z_{i-1} < R_0[s]\}$,
- $BL_{e+1}[s+1] = \{z_i|z_i \notin BL_e[s+1], z_i \in CPS[s] \text{ and } z_{i-1} < R_{e+1}[s]\}$.

Now we have two steps. The first step is devoted to satisfying the requirements, and the second step constructs the c.p.s

Step 1. Suppose $X[s+1] \neq X_L[s]$. Select the least number x so that $X(x)[s+1] \neq X_L(x_L)[s]$. There are two cases.

Case (1). $X(x)[s+1] = 0$. Pick up the least number y which has not yet been ed in the construction. Set $Pre(L)(x_L)[s] <_L y <_L x_L[s]$. Speed up the enumeration of X so that there is a number $z > x$ with $z \in X[s+1]$.

Case (2). $X(x)[s+1] \neq 0$. Suppose $x_L[s] \in (z_i, z_{i+1}]_{\leq_L}[s]$ and $z_{i+1} \in BL_e[s+1]$ for some e. If $x_L[s] > R_e[s]$ then we put $x_L[s]$ into $X_L[s+1]$. If $s+1$ has not been used in the construction, then put $s+1$ into the ordering $<_L$ above all current members of the ordering.

Step 2. We construct CPS by induction on substep i. First, let $CPS[s+1][0] = CPS[s]$. Suppose $y = \max\{z|z \in CPS[s][i]\}$. Search the $\leq_L$-least number $z > y$ so that $\forall z' < z(z' <_L z)$. Define $CPS[s+1][i+1] = CPS[s+1][i] \cup \{z\}$. Finally, set $CPS[s+1] = \cup_i CPS[s+1][i]$.

This finishes the construction.

3.4. **Verification.**

Lemma 3.2. *CPS is a c.p.s.*

Proof. (1) CPS is a computable set. To decide $x \in CPS$ or not. Select a stage $s > x$, then every number $\leq x$ has been included L. So $x \in CPS$ iff $x \in CPS[s]$.

The following facts are easy to see.

(2) $\forall i(z_i < z_{i+1})$.

(3) $\forall z \in \omega \exists i \in \omega(z \leq_L z_i)$. At any stage $s > z$, the $\leq_L [s]$ largest number must be in $CPS[s]$ and so z is L-less than it.

(4) $\forall z \leq z_i(z \leq_L z_i)$.

□

Note we do not claim that CPS is an infinite set although it is. We prove Lemmas 3.3 and 3.4 by simultaneous induction on e.

Lemma 3.3. *N_{e+1} is satisfied and $\lim_s R_{e+1}[s] < \infty$.*

Proof. Select a stage s_0 so that every requirement higher than N_e has been satisfied. Then, by induction (see lemma 3.4), there exists a stage $s_1 \geq s_0$ so that for all stages $t \geq s$, $BL_e[s] = BL_e[t]$ and $\forall z_{i+1} \in BL_e[s]((z_i, z_{i+1}]_{\leq_L}[s] = (z_i, z_{i+1}]_{\leq_L}[t]))$. So there is a stage $s_2 \geq s_1$ so that we will never put any number in those blocks into X_L. Thus, by the construction, the computation in N_{e+1} will be preserved once set up. That means $\forall s, t(s > t \geq s_2 \Rightarrow R_{e+1,s} \geq R_{e+1,t}$ and $\hat{\ell}_{e+1}[s] \geq \hat{\ell}_{e+1}[t])$. Now, by a standard priority argument, N_{e+1} is satisfied and $\lim_s R_{e+1}[s] < \infty$, lets X be computable. □

Lemma 3.4. *There exists a stage s so that for all stage $t \geq s$, $BL_{e+1}[s] = BL_{e+1}[t]$ and $\forall z_{i+1} \in BL_{e+1}[s]((z_i, z_{i+1}]_{\leq_L}[s] = (z_i, z_{i+1}]_{\leq_L}[t]))$.*

Proof. By lemma 3.3, there is a stage s_0 so that $\forall t > s_0(R_{e+1}[s] = R_{e+1}[t] = R_{e+1})$. So, we claim that, $\forall t > s_0(BL_{e+1}[s_0 + 1] = BL_{e+1}[t])$.

Suppose that this does not hold. Select

$$z_{i+1} = \min\{z_{i+1} | z_{i+1} \in BL_{e+1} \text{ and } \lim_s |(z_i, z_{i+1}]_{\leq_L}[s]| = \infty\}.$$

Note z_{i+1} is also $\leq_L$-minimal in BL_{e+1} since $z_i \leq z_{i+1}$ iff $z_i \leq_L z_{i+1}$. Define $Mov(e+1) = \{x | Ord(L)(x) = \infty\}$ and $M(z_{i+1}, e+1) = Mov(e+1) \cap (z_i, z_{i+1}]_{\leq_L}$. Choose $y' \in M(z_{i+1}, e+1)$ so that for every $z \in M(z_{i+1}, e+1)$, $y' \leq_L z$. Actually, $y' \leq R_{e+1}$. To prove this, we prove the following claim:

Claim 3.5. $\forall x \exists y(x \in Mov(e+1) \Rightarrow (y \in Mov(e+1), y \leq R_{e+1}$ *and* $y \leq_L x))$.

Proof. If not, then there is a number $x \in M(z_{i+1}, e+1)$ so that for every y with $y \in Mov(e+1)$ and $y \leq R_{e+1}$, $x <_L y$. Select a stage $s_1 \geq s_0$ so that for every stage $t \geq s_1$ and every number z with $z <_L x$ and $z \leq R$, $Ord(L)(z)[t] = Ord(L)(z)[s_1]$. It follows, that after stage s_1, substep (2.1) in the construction will never apply to any number L-less than x. Define $\hat{M}(x)[s_1] = \{z | z \in (z_i, z_{i+1}]_{\leq_L}[s_1] \cap X_L[t]$ and $z \leq_L x\}$. We enumerate $\hat{M}(x)[s_1] = \{x_i\}_{i \leq n}$ with $x_i <_L x_{i+1}$ for every $i < n$. W.l.o.g, suppose $\forall i(x_i > R(e+1))$. Now, for any number x_i, we move it only in the Construction, Case(1). Suppose we move x_0 at some stage $s_2 \geq s_1$, then by the construction, x_0 is the L-least number so that $X(Ord(x_0))[s_2] \neq X_L(x_0)[s_2 - 1]$. Note we will never move any number L-before x_0 by the construction after stage s_2 since we only apply subcase(2.1) to them. Then by speeding up the enumeration of X, there must be a stage $s_3 \geq s_2$ so that $X(Ord(x_0))[s_3] = 1 = X_L(x_0)[s_3 - 1]$. By the construction, we will never move x_0 after this stage. By induction on i, all of the numbers in $\hat{M}(x)[s_1]$ will stop moving. A contradiction. □

There are only finitely many numbers less than R_{e+1}, so we just need to select the $\leq_L$-least number y' from the set $\{x | x \leq R_{e+1}\}$. Select a stage $s_1 \geq s_0$ so that for every stage $t \geq s_1$ and every number z with $z <_L y'$ and $z \leq R$, $Ord(L)(z)[t] = Ord(L)(z)[s_1]$. By the same reason as in the claim 3.5, we also can select a stage $s_2 \geq s_1$ so that we never apply Case (1) at

step 1 in the construction to any number L-before y'. So by the construction at step 1, we move y' at a stage $t \geq s_1$ only if $Ord(L)(y')[t] \in X_t$. If there are infinitely many such stages, then $\{z | z \geq Ord(L)(y')[s_1]\} \subseteq X$. But X is noncomputable, so coinfinite. A contradiction. □

So $|CPS| = \infty$.

Lemma 3.6. *The built linear order* $\mathbf{L}$ *is isomorphic to* $(\omega, \leq)$ *and and* $X_L = \cup_s X_L[s]$.

Proof. By lemma 3.4, every x will stop moving eventually. It means that for every x, $Ord(L)(x) < \infty$. So $\mathbf{L}$ has order type $\langle \omega, \leq \rangle$ and $X_L = \cup_s X_L[s]$. □

Lemma 3.7. $X_L \leq_T X$.

Proof. For any x, there must be a stage s and a number y with $y = Ord(L)(x)[s]$ so that $\{Ord(L)(z)[s] | z \leq_L x \text{ and } z \in X_L[s]\} = X \upharpoonright (y+1) = \{z | z \leq y \text{ and } z \in X\}$. By the construction, $y = Ord(L)(x)[s] = Ord(L)(x)$. So $x \in X_L[s]$ iff $x \in X_L$. □

Lemma 3.8. X_L *is a low set.*

Proof. It is not hard, by induction on e, to prove that we can effectively in $0'$ find a stage s_e so that $\forall t \geq s_e(\hat{\ell}_{g(e)}[s_e] = \hat{\ell}_{g(e)}[t])$.

By $s-m-n$ theorem, there is a computable function g so that

$$\Phi_{g(e)}^{X_L}(x) = \begin{cases} X(0) & : \ \Phi_e^{X_L}(e) \downarrow \text{ and } x = 0, \\ \uparrow & : \ \text{otherwise.} \end{cases}$$

Then $\Phi_e^{X_L}(e) \downarrow$ iff $\lim_s \hat{\ell}_{g(e)}[s] > 0$. Now we just need to find a stage $s_{g(e)}$ so that $\forall t \geq s_e(\hat{\ell}_{g(e)}[s_e] = \hat{\ell}_{g(e)}[t]$. So $\lim_s \hat{\ell}_{g(e)}[s] > 0$ iff $\hat{\ell}_{g(e)}[s_e] > 0$. Thus $X_L' =_T \emptyset'$. □

4. The Proof of Theorem 1.1

4.1. **The main idea.** The main idea of the proof is somewhat similar to the proof of Theorem 1.3. The property "infinite and coinfinite " is the crucial point again.

Given an infinite and coinfinite c.e. set X, we try to code set Y into the even numbers column of X_L. That is $x \in Y$ iff $2x \in X_L$.

We start by defining the order $\leq_L$ by declaring the order $\leq_L$ on even numbers x and y is consistent with the natural order. That is if x and y are even numbers then $x \leq y$ implies $x \leq_L y$. Then the construction inserts odd numbers into the order.

We ensure that at any even stages s, if $x \in Y[s]$, we put $2x$ into $X_L[s]$. At odd stages s, suppose there is a number x so that $X(x)[s] \neq X_L(x_L)[s]$ at some stage s, then we handle three cases:

- *Case 1*: $X(x)[s] = 0$. Select the least (odd) number y has not been placed in the ordering, define the order $Pre(L)(x_L)[s] <_L y <_L x_L[s]$.
- *Case 2*: $X(x)[s] = 1$ and $x_L[s]$ is odd. Put $x_L[s]$ into $X_L[s]$.
- *Case 3*: $X(x)[s] = 1$ and $x_L[s]$ is even. Select the least (odd) number y has not been put into the ordering, put it into $X_L[s]$ and define the order $Pre(L)(x_L)[s] <_L y <_L x_L[s]$.

As in the proof of theorem 1.3, we can stop moving $x_L[s]$ in the first case by speeding up the enumeration of X since X is infinite. In the second case nothing is moved. In the third case, $x_L[s]$ is moved. But X is coinfinite, so this happens at most finitely often.

In addition, $X_L \leq_T X \oplus Y$ will be true since, given an x, we can effectively in $X \oplus Y$ decide whether x in X_L. Let's turn to the formal proof.

4.2. **Construction.** At stage -1. For all even numbers x, y, declare $x <_L y$ iff $x < y$. Set $X_L[-1] = \emptyset$. We say that every even number has been enumerated into the ordering.

At even stage $2s$, wait (speeding up the enumeration of Y if necessary) for a number $x \in Y[2s] - Y[2s-1]$. Put $2x$ into $X_L[2s]$.

At odd stage $2s+1$, check whether there is a number x so that $X(x)[2s] \neq X_L(x_L)[2s]$. If so, pick up the least one, say x. There are three cases:

(1) $X(x)[2s] = 0$. Select the least (odd) number y has not been enumerated into the ordering, define the order $Pre(L)(x_L)[2s] <_L y <_L x_L[2s]$. Thus, y has been enumerated into the ordering. Speed up the enumeration of X so that there is a number $z > x_L[2s]$ and $X_{2s+1}(z) = 1$.
(2) $X(x)[2s] = 1$ and $x_L[s]$ is odd. Put $x_L[s]$ into $X_L[2s+1]$.
(3) $X(x)[2s] = 1$ and $x_L[s]$ is even. Select the least (odd) number y that has not been enumerated into the ordering, put it into $X_L[2s+1]$ and define the order $Pre(L)(x_L)[2s] <_L y <_L x_L[2s]$. Thus, y has been enumerated into the ordering.

This finishes the construction.

4.3. **Verification.**

Lemma 4.1. *Every number is moved finitely often.*

Proof. It suffices to prove that every even number is moved finitely often. If not, pick up the least such even number, say x. Select a stage s_0 so that $Y[s_0] \upharpoonright x{+}1 = Y \upharpoonright x{+}1$ and no even numbers less than x moved after that. At stage s_0, define $M = \{z \mid \exists y(z = y_L[s_0] \text{ and } x-2 <_L y_L[s_0] \leq_L x \text{ and } X_L(y_l)[s_0] = 1)\}$. Since M is finite, suppose $M = \{z_0 <_L z_1 <_L .... <_L z_n\}$. We prove, by induction on i, that every $z_i \in M$ will stop moving.

Indeed, the element z_0 will stop moving since no number $z <_L z_0$ will move after stage s_0 by the construction. But we move z_0 only when case(1) is applicable. However, by the construction, case (1) will be applied to z_0 only finitely often since we speed up the enumeration of X. Once $X(Ord(L)(z_0))[s] = 1 = X_L(z_0)[s]$ at some stage s then z_0 will stop moving forever since case(1) and case(3) can not be applied to any number L-less than z_0. Select the last stage s_1 at which z_0 moved. Note no number can be inserted between z_0 and x before stage s_1 since we always select the L-least number to do at any stage.

Select the least stage s_{i+1} so that no z_j $(j \leq i)$ moved after that. And, by the induction, no number can be inserted between z_i and x before stage s_{i+1}. Now we can replace z_0 with z_{i+1} in the proof of above to prove z_{i+1} will stop moving eventually.

So, if $Y(x) = 1$ then $x \in M$. Thus x will stop moving eventually. Otherwise, select the least stage $t \geq s_0$ so that no number $z \in M$ moved after that. Note, by the same reason as above, no number can be inserted between z_n and x before stage t. Actually at stage t, we have the following: $\forall z(z_i <_L z \leq_L x \implies X_L(z)[t] = 0)$. This is because some numbers L-less than z_n always require attention before stage t, and hence we have no time to put anything L-between z_n and x into X_L. So, if $X(Ord(L)(x))[t] = 0$, then x will never be moved. Otherwise, we keep on x moving until a stage $t' \geq t$ so that $X(Ord(L)(x)) = X(Ord(L)(x))[t'] = 0$. But X is coinfinite, there must be such a stage. After stage t', x will stop moving.

□

Lemma 4.2. *The linear order* $\mathbf{L}$ *constructed is isomorphic to* $(\omega, \leq)$. *In addition,* $X_L = \cup_s X_L[s]$.

Proof. By Lemma 4.1, every number will stop moving. So $\mathbf{L}$ is isomorphic to $(\omega, \leq_L)$ and $X_L = \cup_s X_L[s]$. □

Lemma 4.3. $Y \leq_T X_L$

Proof. By Lemma 4.2, $X_L = \cup_s X_L[s]$. So by the construction, $x \in Y$ iff $2x \in X_L$. □

Lemma 4.4. $X_L \leq_T X \oplus Y$.

Proof. Take an x. If x is even then $x \in X_L$ iff $\frac{x}{2} \in Y$. If x is odd then select an even number $y >_L x$ and a stage s so that $Y \upharpoonright (x+1) = Y[s] \upharpoonright (x+1)$. By Lemma 4.2, there is a stage $t \geq s$ and a number $y' = Ord(L)(y_L)[t]$ so that $\forall z \leq y'(z \in X[t]$ iff $z_L \in X_L[t])$. So after stage t, y will never been moved. So $x \in X_L$ iff $Ord(L)(x_L)[t] \in X$. □

5. The Proof of Theorem 1.4

For the proof we use Martin's characterization of the high degrees. Say that f *dominates* g if $f(n) \geq g(n)$ for all but finitely many n. In [13] Martin proved that X is *high* ($X' \geq_T \emptyset''$) iff there is an $f \leq_T X$ that dominates all computable functions. Therefore, if X is not high, then for any $f \leq_T X$ there is a computable function g such that $(\exists^\infty n)\ g(n) > f(n)$.

If X is computable then the theorem is clearly true. So, we assume that X is not computable. Hence there are infinitely many bit alternations in X.

Construction. Given a nondecreasing computable function g, which we will choose later, we construct a computable linear order $\mathbf{L}$. During the construction we also declare values of X_L Since we want X_L to be computable, we must continuously adjust $\mathbf{L}$ to ensure that values of X_L, once declared, remain correct. These adjustments could easily prevent $\mathbf{L}$ from being isomorphic to $\langle \omega, \leq \rangle$. This is the purpose of g; we will choose it so that, infinitely often, $X[g(s)]$ has stabilized on a sufficiently large initial segment to ensure that all values of X_L that have been declared by the end of stage $s+1$ are permanently correct without the need of further adjustment to $\mathbf{L}$.

Stage 0. Let $L = \emptyset$.

Stage s+1. Assume that the values of X_L have been declared on $L[s]$. Let y be the $\leq_L$-largest element of $L[s]$. For some $m \leq_L y$, it may be the case that $X(Ord(L)(m))[g(s)] \neq X_L(m)[s]$. These values are currently wrong. Check if there are at least $|L[s]| = Ord(L)(y)[s]+1$ bit alternations in $X[g(s)]$. If so, then it is certainly possible to add elements to $\mathbf{L}$ to ensure that $X(Ord(L)(m))[g(s)] = X_L(m)[s+1]$ for all $m \in L[s]$. Do so, as conservatively as possible. Also add an $x \geq_L y$. Declare $X_L(m)[s+1] = X(Ord(L)(m))[g(s)]$ for all new $m \leq_L x$.

Remark. If the initial segment of $X[g(s)]$ with $|L[s]|$ bit alternations has stabilized, then we will never again add elements to L less than y. So all numbers in $L[s]$ have stopped moving by the end of stage $s+1$. If this happens infinitely often, then $\mathbf{L} \cong \langle \omega, \leq \rangle$ and X_L is computable (all declarations that were ever made are correct).

Defining g. It remains to define the computable function g. First, we define an X-computable function f as follows. To define $f(s)$, consider all possibilities for $L[s]$ over all choices of $g(0), \ldots, g(s-1)$. Note that there are only finitely many possibilities because, for any $i < s$, once $g(i)$ is large enough that $X[g(i)]$ has stabilized on an initial segment with $|L[i]|$ alternations, then no larger value of $g(i)$ would change the resulting $L[i+1]$. Furthermore, X can enumerate all such possibilities because it can detect when sufficient initial segments have stabilized. Let n be the maximum value of $|L[s]|$ over all possibilities and define $f(s)$ so that $X[f(s)]$ has stabilized on an initial segment with at least n bit alternations (and making sure that f is monotone). Then f is an X-computable function.

Because X is not high, there is a computable g, which we used for the construction, such that $(\exists^\infty n)\ g(n) > f(n)$. Hence by the definition of f and the remark above, $\mathbf{L}$ is isomrphic to $(\omega, \leq)$ and X_L is computable. This is the end of the proof.

6. The Proof of Theorem 1.5

Let $\{\langle L_e, W_e, V_e \rangle : e \in \omega\}$ be uniformly effective list, where L_e, $e \in \omega$, is a list of computably enumerable subsets of the rational numbers (hence will be a listing of all computable linear orderings), and (W_e, V_e), $e \in \omega$, is a list of all c.e. pairs of disjoint sets of rational numbers. We need to construct $X \subseteq \omega$ such that if L_e is isomorphic to $\langle \omega, \leq \rangle$, then X_{L_e} is not computable. In the construction we must meet requirements of the following form:

$$Q_e: \text{ If } L_e \cong \omega, \leq \text{ then } X_{L_e} \text{ is not listed by } W_e.$$

We will have a "master" list of numbers $\{x^i : i \in \omega\}$ that we enumerate into X. As we will see below, other numbers may also be enumerated into X. These numbers will be sufficiently far apart for the following to work. For example, we choose x^i to be $f(i)$ for a reasonably fast growing computable function f. We may add numbers into the intervals $(0, x^0)$ or (x^i, x^{i+1}). This will be done in the reverse order: e.g., we would add x^{i+1}-1 first. We allow only Q_0 to enumerate a number into X below x^0, only Q_1 and Q_0 to do this below x_1, etc.

For a fixed e, say $e = 0$, the construction waits till we see a stage s_0 where the pair $(W_e, V_e)[s_0]$ says that the initial segment of $L_e[s_0]$ agrees with the first x^0 many elements of $(\omega, \leq)$. That is, we have $y \in W_e[s_0]$ with $\{z : z <_{L_e} y\} \subset V_e[s_0]$, and all the rational numbers with Gödel numbers $\leq y$ are enumerated into either W_e or V_e at this stage. Thus, it appears that y is the image of x^0. Here we say that x_0 is *realized for* Q_0. (The construction for Q_e will begin with x^e in place of x^0.)

We now enumerate some $p_0 < x^0$ into X_{s_0+1}. The point is that the opponent cannot enumerate any of the current numbers $\{z : z <_{L_e} y\}$ into X_{L_e} since they are already in V_e, and X_{L_e} is supposedly a subset of W_e. Thus, the opponent has to enumerate a new rational number as a potential image of x^0. We conclude that if L_e is isomorphic to $(\omega, \leq)$, and $X_{L_e} \subseteq W_e$, then y is left of the image of x^0 in L_e.

The next action is that $Q_e = Q_0$ would like to move y right of the image of x^1. We wait till we see the initial segment of $(\omega, \leq)[s_1]$ up to some x^1 look the same as the initial segment of L_e of length x^1 at stage $s_1 > s_0$, and this fact is apparently confirmed by W_e and V_e at stage s_1. That is, if there are n elements of X_{s_1} less than or equal to x_1 then there are exactly the same number (and in the same positions) of elements of $L_e[s_1]$ all of which are in W_e and the others are in V_e, and the disjoint union $W_e \sqcup V_e$ computes the appropriate initial segment the natural numbers so that no rational numbers with small Gödel numbers can be enumerated into L_e after stage s_1. Since y has already been declared to be in W_e, we can also ask that y be the apparent image of something in $X[s_1]$. Now one of the following situations will occur for which we describe appropriate actions:

(i) y is already right of the image of x^1 in L_e at stage s_1.
Action: Do nothing, but move on to Q_e processing x^2.

(ii) y is the apparent image of something in $X[s_1] \upharpoonright x^1$.
Remark: If Q_e was the only requirement around, that number would be x^1, but other requirements (such as Q_1) might have already put numbers below x_1 into $X[s]$.
Action: Put into X_{s_1+1} the minimum number of new elements in (x^0, x^1) needed to move y to the right of the image of x^1. Thus if $(x^0, x^1]$ already had k elements in it in X, then we would need to put the greatest k elements not yet in $X[s_1]$ into X_{s_1+1}. (In the actual case of x^1, since only Q_1 could have acted, this number would be either 1 or 2.)

Now the construction proceeds inductively. The goal is for Q_e to drive some $y = y(e)$ to the right of all the potential images of x^i and hence if Q_e acts for all x^i it is not possible for L_e to be isomorphic to $\langle \omega, \leq, X \rangle$ and have witnesses to the computable copy of X in L_e being $W_e \sqcup V_e = \omega$. That is because L_e will have an element $y(e)$ with infinitely many points left of it, so its order type is not ω.

Note that in the interval (x^i, x^{i+1}) since only Q_j for $j \leq i$ will be allowed to enumerate elements into X, it is enough to have the distance between x^i and x^{i+1} be at least $\sum_{j \leq i}(j+1)$.

We now outline the construction above with high permitting. Given a high c.e. set A, we construct a procedure Γ and X meeting the requirements above with $\Gamma^A = X$. We assume by Martin's Theorem that A is enumeration dominant so that for any total computable function g, the principal function of A dominates g. (That is, if $\overline{A}_s = \{a_i^s : i \in \omega\}$ then if $p_A(i) = \mu s[a_i^s = a_i]$, $p_A(i) > g(i)$ for almost all i.)

For a single requirement Q_0, say, the idea is to try to meet Q_0 as above but by using infinitely many potential y_0^n as y's. We then argue that for some n, we meet Q_0 via $y_0^n = y_n$, say. For the sake of Q_0 we will build an auxiliary computable function $g = g_0$.

Initially, we will set $\gamma(i) = a_i^s$ for the procedure below, but this changes in the construction. If we reset $\gamma(i) = a_k^{s+1}[s+1]$ at some stage s, then we would reset $\gamma(j+i) = a_{k+j}[s+1]$ for all j.

We begin as above. For Q_0 we wait till x_0 is realized, and then define $g(0) = s_0$. *We would like to put* x_0-1 *into* X-X_{s_0} *to move* $y = y^0$ *right of the current image of* x^0. *To do this we need a permission from* A *on* $\gamma(0)[s_0]$. Thus, the definition of g challenges the domination of A.

The action at x^0 is *resolved* if at some stage $t_0 > s_0$ A permits $\gamma(0)$. Resolution will allow us to continue the construction. However, whilst we await a resolution at x^0, we start a new strategy based on the belief that no resolution ever occurs at x^0. Thus, we wait for a stage u_1 where till we see the initial segment of $\langle \omega, \leq \rangle[u_1]$ up to some x^1 look the same as the initial segment of L_e of length x^1 at stage $u_1 > s_0$, and this fact is apparently confirmed by W_e and V_e at stage u_1. That is if there are n elements of X_{s_1} less than or equal to x_1 then there are exactly the same number (and in the same positions) elements of $L_e[u_1]$ all of which are in W_e and the others are in V_e, and $W_e \sqcup V_e$ computes the appropriate initial segment the natural numbers so that no rational numbers with small Gödel numbers can be enumerated into L_e after stage u_1. Now we take the apparent image of x^1, y^1, and attempt the construction to satisfy Q_0 using y^1 in the place of y^0.

Thus, we now define $g(1) = u_1$, and await an A permission on $\gamma(1)[u_1]$. We remark that all strategies for Q_0 will be using the *same* computable g. We continue as above inductively, starting strategies at various x^i using y^i.

Say, at some stage s_i we get a permisssion on some $\gamma(i)$. Then in the interval (x^{i-1}, x^i) we put the necessary number of elements into X to move y^i right of the apparent image of x^i. Next, we will make the use of $\Gamma^A(i+k)[s_i+1]$ bigger. That is, we find a large fresh number d and set $\gamma(i+k, s_i+1) = a_{p+i+k,s_i+1}$ for all $k > 1$. The action is not that all lower priority strategies for Q_0 are initialized at this stage. This entails that a strategy pursuing x^j on the assumption that the i-strategy is stuck is now initialized *forever*, and we would only pursue m-strategies for Q_0 using $m \leq i$ and $m \geq s_i$. The point of "kicking" the uses for $\gamma(i+1, s_i+1)$ is the following. What the i-strategy for Q_0 would like to move y^i right of the image of x^{i+1}. For this it needs a permission from A. We would like to argue that if we ever fail then A is not dominant. But some other strategy migh have already defined $g(i+1)$. Now, A might have fulfilled its commitment for dominating $g(i+1)$ and may not

actually care to change again, *after* we are ready to move y^i again. Since we have kicked the $i+1$-use to a place that we have not as yet defined g, we can allow that i-strategy to assert control of $g(p+i+1)$. That is, we next wait for L_0 to give us an apparent isomorphism for L_e up to x^{i+1} again, but now at that stage s_2 we will define $g(p+i+1)$ to try to pursue the i-strategy. Of course, the other i'-strategies for $i' < i$ might act before this, initializing (forever) the i-strategy, but there is some i which is immortal. Notice that g is total if we fail to meet Q_0 and would not be dominated by p_A, a contradiction. Also $X \leq_T A$ by permitting on γ.

Notice also that, even for a single Q_0, we would need the intervals (x^i, x^{i+1}) to be larger, since differing Q_0 strategies can enumerate elements into these intervals for their own y^i's.

Now we consider the combination of the requirements. Consider Q_1. The outcomes for the procedure above have order type $\omega + 1$, namely $0, 1, 2, \ldots, f$, where f denotes that Q_0 fails to have infinitely many stages where L_e can be considered as isomorphic to $(\omega, \leq, Q)$ with witness $W_0 \sqcup V_0$, and the others denote the least n where the n-strategy acts infinitely, and hence succeeds.

In a standard way, each outcome has its own version of Q_1. Q_1^f guessing f will simply work on x^i which are untouched by Q_0. Q_1^n guessing n will "know" that n is the correct outcome, and builds its versions of g_1, g_1^n at stages when n looks correct. There are little interactions between the requirements to ensure that $\gamma(i, s)$ has a limit; this also happens in a standard way. The remaining details are a typical tree of strategies argument, and provide little insight but detail.

7. The Proof of Theorem 1.6

This is a short proof. Given a c.e. set $X \subseteq \omega$ we construct a set Y m-equivalent to X such that Y has a computable copy in a computable copy of $(\omega, \leq)$. The idea is simple. We define a "very spread out and smeared" version of X, so that no point can be pushed to infinity. Thus break the natural numbers into long intervals I_k for $k \in \mathbb{N}$, and interval I_k is has coing locations for $j \leq k$ as follows. The coding location for k is the first element, then there are large (ever increasing) gaps with coding locations for $j < k$ in increasing order, and with large gaps between them. Let $c(j, k)$ denote the coding location for j in interval k.

We will build Y in the primary copy of $(\omega, \leq)$ using only the coding locations. We declare that $j \in X$ iff $c(j, k) \in Y$ for all j. Then clearly $Y \equiv_m X$

The computable copy of Y we build is made by the least effort strategy. Namely, we will have an ordering $\leq_A$ at stage s, and an isomorphism from $\langle \omega, \leq, Y \rangle[s]$ to $A = \langle \omega, \leq_A, Y_A \rangle[s]$. If some element j enters $Y_{s+1} - Y_s$, putting coding markers into all intervals I_k for $k \geq j$, then we add new elements from $\mathbb{N} - \mathrm{dom}(A_s)$. to regain the isomorphism, moving the image of each point as little as possible (at worst to the stage s image of the least point in Y_s above it). Clearly the copy of Y in A must be computable if it exists. By induction, no point in A which is the image of a point in I_k can ever be moved out of

the image of $A \upharpoonright (\min\{p : p \in I_{k+2}\} - 1)$, and so the isomorphism exists. This ends the proof.

8. The Proof of Theorem 1.8

For the proof of the theorem we need a series of lemmas. The first lemma is an easy exercise:

Lemma 8.1. *Let X_L be a copy of $X \subseteq \omega$ in a computable linear order $\mathbf{L}$ isomorphic to $(\omega, \leq)$. Then $X_L \leq_T X \oplus \emptyset'$ and $X \leq_T X_L \oplus \emptyset'$. Particulary, if both X_L and X compute $\emptyset'$, then $X \equiv_T X_L$.* □

Consider the modulus function of the halting problem K: $f(n) = \mu(s)(K_s \upharpoonright n = K \upharpoonright n)$. Define a set $S = \{f(n) | n \in \mathbb{N}\}$. The next lemma is again easy.

Lemma 8.2. *Given a set $X = \{x_1 < x_2 < ... < x_n < ...\}$ which satisfies the following property:*

$$\exists m \forall n \geq m (a_n \geq f(n)).$$

Then $X \geq_T K$.

Note if X has the property in Lemma 8.2, then every infinite subset of X has the property too. The proof of the next lemma is more involved.

Lemma 8.3. *There is a Δ^0_2-set X so that each copy X_L of X in computable presentation $\mathbf{L}$ of $(\omega, \leq)$ has the property in Lemma 8.2.*

Proof. We construct a Δ^0_2-set X which has the property in Lemma 8.2. Fix an effective enumeration of computable linear orderings $\mathbf{L}_e = (\omega, \leq_{L_e})$. Particularly, define $\mathbf{L}_0 = (\omega, \leq)$.

To decide x_e, we look at $\{\mathbf{L}_i\}_{i \leq e}$. This number needs to be one so that number $x > \sum_{i<e} x_i$ chosen so that $x_{L_i} > f(e)$ for each $i \leq e$. The problem is that we cannot actually decide what a_{L_i} from $\emptyset'$, since $\mathbf{L}_e$ might not even be isomorphic to $\mathbf{L}$. But we can $\emptyset'$-effectively find a number a for which $x_{L_i} > f(e)$ for each $i \leq e$.

This is done using a recursive procedure as follows. For a fixed i and $f(e)$ we can definitely $\emptyset'$ decide if it is *i-bad* (for $f(e)$). That is, $x_{L_i} \leq f(e)$. Namely for a fixed x we can run the enumeration of $\mathbf{L}_i$ and (using $\emptyset'$) discover that s is a stage where $x_{L_i}[s] = x_{L_i}$ and $x_{L_i} \leq f(e)$. We claim that this is a basis for finding an x which is *i-good.* Take some x, run the enumeration of $\mathbf{L}_i$ until we discover that either x is i-bad, in which case we pick a new x and try again with this x, or we get a certification that $x_{L_i}[s] = x_{L_i}$ and $x_{L_i} > f(e)$, or we reach a stage s where for all $y \leq f(e)$,

(i) either $y = b_{L_i}[s] = b_{L_i} \neq x_{L_i}$, or
(ii) $x_{L_i} <_{L_i} y$. (That is y has moved beyond x_{L_i}.)

Then we can see that x is i-good since x_{L_i} cannot be i-bad. Since there are only at most $f(e)$ many i-bad elements for each i, we can run the procedures above for each $i \leq e$ together and hence find an element x which is i-good for $f(e)$ for all $i \leq e$. (Alternatively we could find $ef(e) + 1$ many numbers which are 1-good for $f(e)$, of these at least $(e-1)f(e) + 1$ must be 2-good for $f(e)$, etc.)

This allows us to define x_e and we put this into A.

If $\mathbf{L}$ is isomorphic to $(\omega, \leq)$ and is the e-th linear ordering, $(x_e)_L \geq f(e)$ for each e by the construction. Rearrange X_L so that $X_L = \{b_1 < b_2 < ...b_n < ...\}$. There is a bijection $g : \omega \to \omega$ so that $b_e = (x_{g(e)})_L$. Take the first $i \leq e$ so that $g(i) \geq e$. Note there must be such an i.

Now we prove, by induction on j, that $b_{i+j} \geq f(i+j)$. For $j = 0$, $b_i = (x_{g(i)})_L \geq f(g(i)) \geq f(e) \geq f(i)$. For $j > 0$, $b_{i+j} = (x_{g(i+j)})_L$. If $g(i+j) \geq i+j$, then $b_{i+j} = (x_{g(i+j)})_L \geq (x_{i+j})_L \geq f(i+j)$. Otherwise, there must be some $k < i+j$ for which $g(k) \geq i+j$. Then $b_{i+j} > b_k = (x_{g(k)})_L \geq f(g(k)) \geq f(i+j)$. □

Lemma 8.4. *For each set $B \geq_T \emptyset'$, there is a set $X \equiv_T B$ so that for each copy X_L of X, $X_L \equiv_T B$.*

Proof. Take a Δ^0_2 set $C = \{c_1 < c_2 < ... < c_n...\}$ as in Lemma 8.3. Define $X = \{c_n | n \in B\}$. Then each copy X_L of X satisfies the property in Lemma 8.2 and so computes $\emptyset'$. Hence X_L computes C_L for each L. Particulary X computes C and so computes B. Obviously B computes X. So, $B \equiv_T X$. By Lemma 8.1, each $X_L \equiv_T X \equiv_T B$. □

These lemmas prove the theorem.

9. The Proof of Theorem 1.9

For the proof we need to construct a Δ^0_2-set X so that for each e the set X_e is low. Thus we need to meet the requirements:

$$N_{e,i} : \text{If } \mathbf{L}_e \cong \mathbf{L} \text{ then } \exists^\infty s \Phi_i^{X_{e,s}}(i) \downarrow \to \Phi_i^{X_e}(i) \downarrow,$$

where $\mathbf{L}$ is the structure $(\omega, \leq, X)$ that we are building. Additionally, we must ensure that X has no computable copy. Thus we also need to meet the requirements:

$$P_{e,i} : \text{If } \mathbf{L}_e \cong \mathbf{L} \text{ then } X_e \neq \varphi_i.$$

We meet $P_{e,i}$ in the standard way. In L_e we pick a witness $x = x(e,i)$. We wait for a stage s where $\varphi_i(x) \downarrow$. At such a stage we commit to putting $x \in X_e$, or keep x out of X_e to make sure that $X_e(x) \neq \varphi_i(x)$. Let y_s denote the element of $\mathbf{L}$ corresponding to x at stage s.

For simplicity lets suppose that $\varphi_i(x) = 0$. Then this commitment will ask that we put $y_s \in X_s$. We say that $P_{e,i}$ *asserts control* of y_s. At a later stage t it might be that $y_t \neq y_s$ since a new element has entered $\mathbf{L}_e$ $<_{L_e}$-below x. At this stage t, $P_{e,i}$ releases control of y_s and asserts control of y_t, by putting it into X_t. Note that once $P_{e,i}$ releases control of y_s if will never again assert control of y_s. If $\mathbf{L}_e \cong \mathbf{L}$ then $\lim_s y_s$ exists and we will succeed in meeting $P_{e,i}$.

The only thing that we ensure is that the witnesses are chosen $<_{L_e}$ monotonically. Thus, for $i < j$, if we choose x for $P_{e,i}$ then we can only choose some x' for $x <_{L_e} x'$ as a follower for $P_{e,j}$. This can be done in case $\mathbf{L}_e \cong \mathbf{L}$. If $\mathbf{L}_e \ncong \mathbf{L}$ it might be possible that only finitely many followers might be chosen for requirements of the form $P_{e,d}$. The reason for this is the following. If $\mathbf{L}_e \ncong \mathbf{L}$, because some $n \in \mathbf{L}_e$ has infinitely many $<_{L_e}$ predcessors, then

should we happen to choose some x where y_s changes infinitely many times, then almost all of the y'_s will also be driven to infinity. This means that there effect on X will be "transitory."

Now we turn to the method of meeting the $N_{e,i}$. At some stage s, we might see $\Phi_i^{X_{e,s}}(i)$. Then we would pursue the usual method of lowness. That is $N_{e,i}$ would assert control of $u(e,i,s) = X_{e,s} \upharpoonright \varphi_i^{X_{e,s}}(i)$. It would ask that this set be preserved forever, and thus we would win.

Again this would entail looking at the pre-image of $u(e,i,s)$ in $\mathbf{L}$ and asking that they not change their status with respect to X. This will mean that there is a set of numbers $z(e,i,s)$ which correspond to the use $u(e,i,s)$ in $\mathbf{L}_s$ whose membership pattern is determined by $\Phi_i^{X_{e,s}}(i)$. At stage s control will be asserted with priority $N_{e,i}$, upon $\Phi_i^{X_{e,s}}(i)$ For a single $N_{e,i}$ if $\mathbf{L}_e \cong \mathbf{L}$, these pre-images will eventually settle down, and we will succeed in meeting $N_{e,i}$, with finite effect. In the case that $\mathbf{L}_e \ncong \mathbf{L}$, it might be possible that some $\leq_{L_e}$-least member z of $z(e,i,s) \to \infty$. Then a finite number of $\mathbf{L}$ will be preserved at almost all stages, but the overall eeffect of $N_{e,i}$ will be that those numbers $\leq_{L_e}$ above z below $u(e,i,s)$ will have only transitory effect on X.

Now we need to consider the overall effect of the interactions of the requirements. The first effect is that various higher priority $P_{f,j}$ will be able to injure some $N_{e,i}$. As outlined above some $P_{f,j}$ will be able to assert control over some $y = y_s$ in $\mathbf{L}$, to ask that it enter (or leave) $X_{e,s}$. All is sweet unless y corresponds to some pull back of an element in the use $u(e,i,s)$.

Now the plan is to allow $P_{f,j}$ to injure $N_{e,i}$ if it has higher priority by simply letting it do what it wants. If it injures $N_{e,i}$ then we would need to find a new $\mathbf{L}_e$ configuration for $N_{e,i}$ to preserve, with its own priority. Whilst we are doing this we will remember this desirable $N_{e,i}$ configuration. It may be that this injury by $P_{f,j}$ might be f,j-transitory, in that $y_s \to \infty$, since $\mathbf{L}_f \ncong \mathbf{L}$.. Once y_s clears the $\mathbf{L}$-zone corresponding to the $u(e,i,s)$ elements, then we would be free to return to that configuration. Note that, thereafter, *unless* $\mathbf{L}_e$ itself changes the region of $\mathbf{L}$ that it needs to control to preserve the $u(e,i,s)$-computation, $P_{f,j}$ will have no further efect on $N_{e,i}$. Thus in this case, *provided that* $\mathbf{L}_e \cong \mathbf{L}$, if the action of $P_{f,j}$ is infinitary, it basically has only finite effect on $N_{e,i}$.

On the other hand, it might well be possible that $P_{f,j}$ might be concerned with some $\mathbf{L}_f \cong \mathbf{L}$ and hence the position of y_s might settle down, and might have a permament injury to $N_{e,i}$. This case is handled in the obvious way. When $N_{e,i}$ was injured by $P_{f,g}$ as above, then we would look for a new computation involving $\Phi_i^{X_{e,s'}}(i)$ for some $s' > s$ to preserve. In the situation that $P_{f,g}$ settles down, the next configuration would be one that could really be preserved with priority e,i.

Similar comments pertain to $N_{q,r}$ of higher priority than $N_{e,i}$ which again might have their own demands as to the look of X in the region that $N_{e,i}$ wishes to control. Again we note that either this effect is permamant and we would meet $N_{e,i}$ after this injury, or we would be able to resurrect the $u(e,i,s)$ computations once the $N_{q,r}$ demands have permamantly cleared the use.

Clearly, arguing by priorities, we would not allow a lower priority $P_{g,d}$ to injure some $N_{e,i}$. If the relevant x had a y_s which would be in some critical region corresponding to $N_{e,i}$ then we would regard this as a (possibly) temporary injury to $P_{g,d}$. and we would be able to meet $P_{g,d}$ with another follower. The same considerations hold for $N_{r,s}$ of lower priority. Really there is guessing going on here, since for any fixed r we could know the outcome of the $\mathbf{L}_e$ for $e < r$, but if $\mathbf{L}_r \cong \mathbf{L}$ then all injuries are finite, and hence we will be able to argue that all the requirements are met. The remaining details are straightforward.

10. The Proof of Theorem 1.10

For the proof of the theorem we need to exhibit two computable presentations $\mathbf{L}_1$ and $\mathbf{L}_2$ of $(\omega, \leq)$. The definitions of $\mathbf{L_1}$ and $\mathbf{L_2}$ are quite simple. For $\mathbf{L_1}$, first let L_1 be the even numbers under the usual ordering. If n goes into $\emptyset'$, put the least remaining odd number immediately *before* $2n$. For $\mathbf{L_2}$, again let L_2 be the even numbers under the usual ordering. But now if n goes into $\emptyset'$, put the least remaining odd number immediately *after* $2n$.

Assume that X is infinite and coinfinite. To a large extent, the blocks of bits in X are reflected in $X_{L_1} \upharpoonright 2\omega$. The only exception is when X has an isolated bit at position x and x_{L_1} is odd. But if x_{L_1} is odd, then $(x+1)_{L_1} = x_{L_2}$ is even. So $X_{L_1}(x_{L_2}) = X_{L_1}((x+1)_{L_1}) = X(x+1) \neq X(x) = X_{L_2}(x_{L_2})$ and thus we can detect the missing block by looking for a discrepancy between $X_{L_1} \upharpoonright 2\omega$ and $X_{L_2} \upharpoonright 2\omega$.

Once we have located the missing blocks of bits, it is straightforward to compute $\emptyset'$. To determine if $n \in \emptyset'$, find an even $m > 2n$ such that either $X_{L_1}(m) \neq X_{L_1}(m-2)$ or there is a "missing block" before m. Run the construction of $\mathbf{L_1}$ until a stage s such that $X_{L_1}[s] \upharpoonright m+1$ agrees on the even bits with $X_{L_1} \upharpoonright m+1$ *and* every missing block has been accounted for. In other words, if we determined that there is a missing block in $X_{L_1} \upharpoonright 2\omega$ right before $2t \leq m$, then the immediate predecessor of $2t$ in $L_1[s]$ must be odd. Once such a stage s has been reached, no number $\leq n$ can enter $\emptyset'$ because otherwise we would exceed the number of bit alternations—missing and apparent—in $X_{L_1} \upharpoonright 2\omega$ before position m. Therefore, $n \in \emptyset'$ iff $n \in \emptyset'_s$. We have proved the theorem.

References

[1] C. J. Ash and A. Nerode, Intrinsically recursive relations, in J. N. Crossley (ed.), Aspects of Effective Algebra (Clayton, 1979) (Upside Down A Book Co., Yarra Glen, Australia, 1981) 26–41.

[2] R. G. Downey, Computability theory and linear orderings, in Y. L. Ershov, S. S. Goncharov, A. Nerode, and J. B. Remmel (eds.), Handbook of Recursive Mathematics, vol. 138–139 of Stud. Logic Found. Math. (Elsevier Science, Amsterdam, 1998) 823–976.

[3] Downey, R. G., S. Lempp, and G. Wu, On the complexity of the successivity relation in computable linear orderings, in preparation.

[4] Downey, R. G. and M. F. Moses, Recursive linear orderings with incomplete successivities, Trans. Amer. Math. Soc., vol. 320 (1991), 653–668.

[5] Harizanov, V., Some effects of Ash–Nerode and other decidability conditions on degree spectra, Ann. Pure Appl. Logic. vol. 54 (1991), 51–65.

[6] Harizanov, V., Uncountable degree spectra, Ann. Pure Appl. Logic. vol. 54 (1991), 255–263.
[7] Harizanov, V., Turing degree of the non-zero member in a two element degree spectrum, Ann. Pure and Appl. Logic, vol. 60 (1993), 1–30.
[8] Hirschfeldt, D., *Degree Spectra of Relations on Computable Structures*, PhD Thesis, Cornell University, 1999.
[9] Hirschfeldt, D., Miller, R., and Podzorov S. Order Computable sets. Notre Dame Journal of Formal Logic, no 3 (2007), p.317-347.
[10] Goncharov, S. S., On the number of nonautoequivalent constructivizations, Algebra and Logic, Vol. 16 (1977), 169–185.
[11] Hirschfeldt, D., B. Khoussainov, R. Shore and A. Slinko, Degree spectra and computable dimension in algebraic structures,
[12] B. Khoussainov and R. A. Shore, Computable isomorphisms, degree spectra of relations, and Scott families, Ann. Pure Appl. Logic. vol. 93 (1998) 153–193.
[13] D. A. Martin., Classes of recursively enumerable sets and degrees of unsolvability, Z. Math. Logik Grundlag. Math. vol. 12 (1966), 295–310.
[14] Moses, M. F., Relations intrinsically recursive in linear orderings, Z. Math. Logik Grundlagen. Math., vol. 32 (5), 467–472.
[15] R. I. Soare, Recursively Enumerable Sets and Degrees, Perspect. Math. Logic (Springer–Verlag, Heidelberg, 1987).

Rod Downey, School of Mathematical and Computing Sciences, Victoria University, PO Box 600, Wellington, New Zealand
E-mail address: Rod.Downey@mcs.vuw.ac.nz

Bakhadyr Khoussainov, Depart of Computer Science, Auckland University, Private Bag 92019, Auckland, New Zealand
E-mail address: bmk@cs.auckland.ac.nz

Joseph S. Miller, Department of Mathematics, University of Connecticut, Storrs, CT 06269-3009, USA
E-mail address: joseph.miller@math.uconn.edu

Liang Yu, Institute of Mathematical Science, Nanjing University, JiangSu province 210093, China
E-mail address: yuliang.nju@gmail.com