Degree Spectra of Unary Relations on ω and ζ

by

Carolyn Knoll

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Pure Mathematics

Waterloo, Ontario, Canada, 2009

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Let X be a unary relation on the domain of $(\omega, \leq)$. The degree spectrum of X on $(\omega, \leq)$, denoted $DgSp_{(\omega,\leq)}(X)$, is the set of Turing degrees of the image of X in all computable presentations of $(\omega, \leq)$. Many results are known about the types of degree spectra that are possible for relations forming infinite and coinfinite c.e. sets, high c.e. sets and non-high c.e. sets on the standard copy. We show that if the degree spectrum of X contains the computable degree then its degree spectrum is precisely the set of Δ_2 degrees.

The structure ζ can be viewed as a copy of ω^* followed by a copy of ω and, for this reason, the degree spectrum of X on ζ can be largely understood from the work on $(\omega, \leq)$. A helpful correspondence between the degree spectra on $(\omega, \leq)$ and ζ is presented and the known results for degree spectra on the former structure are extended to analogous results for the latter.

Acknowledgements

I would like to thank my advisor, Professor Barbara Csima, for her countless hours of reading and her unwavering guidance through this new topic. I would also like to thank my readers, Professor Ross Willard and Professor John Lawrence, for their many insights and suggestions.

Dedication

To J. Wearing and S. Knoll for their love and support from across the country, and to my sister Jenny, for "allowing" me to discuss my studies at the dinner table.

Contents

Chapter 1

Introduction

In mathematics we often classify structures up to isomorphism, as for our purposes, they are identical. In this paper we will examine isomorphic models from the point of view of computable structure theory. While isomorphic structures can often be identified, they can exhibit different computability theoretic behaviour. We will explore these differences by considering the complexity of the same relation on various isomorphic copies of a given structure. In particular, we will look at the collection of Turing degrees of images of a unary relation in different computable presentations of $(\omega, \leq)$ and $(\omega, \leq_\zeta)$, the standard copy of the natural numbers $\mathbb{N}$ and the integers $\mathbb{Z}$.

Before we begin, we need to familiarize ourselves with the notion of a computable structure, relations on the given structure, and what we mean by a computable presentation of a structure ([4]).

Definition. *A structure $\mathcal{A}$ is computable if its domain $A = |\mathcal{A}|$, basic relations and basic functions are uniformly computable.*

In this paper we are concerned with particular linear orderings, so for example, a linear ordering $\mathcal{L} = (L, \leq_\mathcal{L})$ is computable if the universe L and the order relation $\leq_\mathcal{L}$ are both computable.

Definition. *An isomorphism φ from a structure $\mathcal{A}$ to a computable structure $\mathcal{B}$ is*

called a computable presentation of $\mathcal{A}$.

We will abuse the above notation and refer to the image of $\mathcal{A}$ under φ, namely $\mathcal{B}$, as a computable presentation or a computable copy of $\mathcal{A}$. In the case of a linear ordering $\mathcal{L}$, a computable copy of $\mathcal{L}$ is an isomorphic image where the order relation is computable. As mentioned above, we will be studying the behaviour of a unary relation R on a structure $\mathcal{A}$ as we range over all computable copies of $\mathcal{A}$.

Definition. *Let R be a relation on the domain of a structure $\mathcal{A}$. Let $\mathcal{B}$ be a computable copy of $\mathcal{A}$ with $\varphi : \mathcal{A} \cong \mathcal{B}$.*

1. *We define $R_{\mathcal{B}}$ to be the image of R in $\mathcal{B}$ (under φ).*

2. *The degree spectrum of R on $\mathcal{A}$, denoted $DgSp_{\mathcal{A}}(R)$, is defined to be the set of (Turing) degrees of $R_{\mathcal{B}}$ as we range over all computable presentations $\mathcal{B}$ of $\mathcal{A}$.*

With these notions in hand, we can begin to explore different properties of degree spectra. Are there choices of $\mathcal{A}$ and R where $DgSp_{\mathcal{A}}(R)$ is trivial, is a singleton, has an upper or lower bound, is upward closed in the (Turing) degrees? (I.e. If $\mathbf{a} \in DgSp_{\mathcal{A}}(R)$ and $\mathbf{b} \geq \mathbf{a}$ then $\mathbf{b} \in DgSp_{\mathcal{A}}(R)$.)

Now we are ready to focus our attention on the degree spectra of relations on our structures of interest: $(\mathbb{N}, \leq_{\mathbb{N}})$ and $(\mathbb{Z}, \leq_{\mathbb{Z}})$. These are of course the natural numbers and the integers in their standard order. As we are dealing with computable structures in this paper, we will identify the domain of each structure with ω under a clear coding. We will denote the natural numbers as $(\omega, \leq)$, having domain ω under the $\leq_{\mathbb{N}}$-order. We will denote the integers as $(\omega, \leq_\zeta)$, again having domain ω, but under the $\leq_{\mathbb{Z}}$-order. The two structures, under our coding, are given below:

$(\omega, \leq)$ $\qquad$ 0 1 2 3 4 . . .

$(\omega, \leq_\zeta)$ $\qquad$ . . . 7 5 3 1 0 2 4 6 8 . . .

These two orderings are of course computable structures since the order relations are computable, but moreover, the binary adjacency relation $S(x, y)$ in each is computable. We wish to ask how a given relation on either structure can change as we range over all computable copies of the structure. In other words, we wish to determine the degree spectrum of this relation. To get us started on studying degree spectra, we will give an intuitive approach to determining the degree spectra of a well studied relation: the adjacency relation, S, on $(\omega, \leq)$.

The structure $(\omega, \leq)$ is rigid, (i.e. it has only one order-preserving automorphism) but the complexity of a relation on $(\omega, \leq)$ can change when considering its image in a different computable presentation of $(\omega, \leq)$. It is not difficult to build a computable copy of the natural numbers in which the adjacency relation is not computable. From this remark, we know that $DgSp_{(\omega,\leq)}(S)$ is non-trivial. It turns out that this relation is an example in which the degree spectra is bounded above. The reasoning is as follows: Let $\mathbf{L}$ be a computable copy of $(\omega, \leq)$. Since the order relation $\leq_{\mathbf{L}}$ is computable, we can decide whether $x \leq_{\mathbf{L}} y$ for any two elements x and y in the domain of $\mathbf{L}$. If we compute that x is $\leq_{\mathbf{L}}$-less than y in our ordering, then at that moment we believe that the pair $(x, y) \in S$. We will believe this until we find a $z \in \omega$ that lies strictly between our given points. The instant we see $x \leq_{\mathbf{L}} z \leq_{\mathbf{L}} y$, for some z, we will declare with absolute certainty that $(x, y) \notin S$. Thus the complement of S is c.e. by nature. Since any co-c.e. set is $\emptyset'$-computable, $DgSp_{(\omega,\leq)}(S)$ is bounded above by $\mathbf{0}'$.

In this communication we will restrict ourselves to unary relations, X, which can be viewed as subsets of the domain ω. As $X \subseteq \omega$, $DgSp_{(\omega,\leq)}(X)$ and $DgSp_{(\omega,\leq_\zeta)}(X)$ will contain all Turing degrees of the image of X in each computable copy of $(\omega, \leq)$ and $(\omega, \leq_\zeta)$, respectively. Again, since we are only dealing with orderings with countable domains, we will identify the domain of any relevant copy with ω. So we will denote a copy of $(\omega, \leq)$ or $(\omega, \leq_\zeta)$ as $\mathbf{L} = (\omega, \leq_{\mathbf{L}})$ where $\leq_{\mathbf{L}}$ is an binary order relation and $\mathbf{L} \cong (\omega, \leq)$, or $\mathbf{L} \cong (\omega, \leq_\zeta)$.

Lastly, before we begin, we need a bit of notation, some taken from [2], that we will be using throughout the paper in our proofs.

1.1 Notation and Conventions

.

Let $\mathbf{L} = (\omega, \leq_{\mathbf{L}})$ be a computable copy of $(\omega, \leq)$ or $(\omega, \leq_\zeta)$ under some isomorphism φ.

- $\mathbf{L}\upharpoonright x$ denotes the intitial segment of $\mathbf{L} <_{\mathbf{L}} x$, and $\mathbf{L}\restriction\uparrow x$ denotes the segment of $\mathbf{L} \leq_{\mathbf{L}} x$.
- For each $x \in \omega$, let $x_{\mathbf{L}} = \varphi(x)$, the image of x in $\mathbf{L}$.
- Let $X_{\mathbf{L}} = \varphi(X)$, the image of the set X in $\mathbf{L}$.
- In a stage by stage construction, an application of $[s]$ means that each stage dependent member of the statement is evaluated at stage s. For example we write $\Gamma^A(x) = X(x)[s]$ for $\Gamma_{A_s}(x) = X_s(x)$ and $\langle L, W, V\rangle[s]$ for $\langle L_s, W_s, V_s\rangle$.

Chapter 2

Degree Spectra on $(\omega, \leq)$

In this section we will discuss what is known about the degree spectra of unary relations on the computable structure $(\omega, \leq)$. An in depth look at the subject can be found in a pre-paper entitled "Degree Spectra of Unary Relations on $(\omega, \leq)$" by R. Downey, B. Khoussainov, J. Miller and L. Yu ([2]) and "Order-computable Sets" by D. Hirschfeldt, R. Miller and S. Podzorov ([3]). The results presented here will provide an overview of what kinds of degree spectra are possible on this structure.

2.1 Degree Spectra of c.e. Sets

We will begin by discussing unary relations on the structure $(\omega, \leq)$ that form c.e. sets on the standard copy. We wish to determine whether the degree spectra of these sets have a lower bound or indeed contain the computable degree. We can also ask whether the degree spectrum of such a set X is upward closed in the Turing degrees, or whether we can determine an upper bound. We begin with a look at a result from [2] exploring the degree spectrum of an infinite and coinfinite c.e set.

Theorem 2.1 (Downey, Khoussainov, Miller and Yu). *Let X be an infinite and coinfinite c.e. set and let Y be an infinite c.e. set such that $X \leq_T Y$. Then there is a computable presentation $\boldsymbol{L}$ of $(\omega, \leq)$ such that $X_{\boldsymbol{L}} \equiv_T Y$.*

The idea of the proof is as follows. Given the sets X and Y, we will construct a computable copy $\mathbf{L}$ of $(\omega, \leq)$ while using the even numbers of $X_{\mathbf{L}}$ to code the set Y. To begin, we will fix enumerations of X and Y, say $\{X_s\}_{s\in\omega}$ and $\{Y_t\}_{t\in\omega}$. At stage -1 we will declare all of the even numbers into our ordering $\mathbf{L}$ in their natural $\leq_{\mathbb{N}}$-order, and enumerate $2x$ into $X_{\mathbf{L}}$ at stage s if and only if x goes into Y at stage s. We then use the odd numbers to ensure that $X_{\mathbf{L}}$ is indeed the isomorphic image of X in $\mathbf{L}$. If at stage s we see that $X(z)[s] \neq X_{\mathbf{L}}(z_{\mathbf{L}})[s]$ for some $z \in \omega$, then we insert the least (odd) number not yet in $\mathbf{L}$ in order to fix the discrepancy. We may not immediately fix the problem by enumerating only one new number, but we will succeed after repeating this action finitely many times.

A full proof of this result is found in [2] and so is not presented here. The proof relies on the fact that the set X is infinite and coinfinite which ensures that, in the end, every number in $\mathbf{L}$ settles and $\mathbf{L} \cong (\omega, \leq)$ as desired. We have $Y \leq_T X_{\mathbf{L}}$ since $x \in Y$ if and only if $2x \in X_{\mathbf{L}}$. We can compute $X_{\mathbf{L}}$ from X and Y and so $X_{\mathbf{L}} \leq_T X \oplus Y \equiv_T Y$ since $X \leq_T Y$ by assumption. Hence we have $X_{\mathbf{L}} \equiv_T Y$ as desired.

Theorem 2.1 shows that if a unary relation X forms an infinite and coinfinite c.e. set on $(\omega, \leq)$ then $deg(Y) \in DgSp_{(\omega,\leq)}(X)$ for all c.e. sets $Y \geq_T X$. It follows that if this set X is computable on the standard copy, not just c.e., then $DgSp_{(\omega,\leq)}(X)$ contains all of the c.e. degrees. It is an interesting question to ask whether or not we get a similar result for an infinite and coinfinite set X that is computable on some arbitrary computable copy of $(\omega, \leq)$, not necessarily the standard copy. This result will be explored in the next section.

2.2 Degree Spectra Containing 0

A set whose degree spectrum contains the computable degree is called *order-computable* [3]. We wish to determine whether the degree spectrum of an order-computable set X contains all c.e. degrees.

Clearly, any finite or cofinite set X is computable on the standard copy, and hence is order-computable, but the image of X in any computable copy $\mathbf{L}$ must again be

finite or cofinite. Thus in this case, $DgSp_{(\omega,\leq)}(X)$ does not contain all c.e. degrees, in fact $DgSp_{(\omega,\leq)}(X) = \{\mathbf{0}\}$. Let us assume that X is an infinite and coinfinite set. It turns out that if X is order-computable, then $DgSp_{(\omega,\leq)}(X)$ does indeed contain all c.e. degrees. Moreover, we can show that $DgSp_{(\omega,\leq)}(X)$ contains all Δ_2 degrees.

By Post's Theorem, we have that a set $A \in \Delta_2$ if and only if $A \leq_T \emptyset'$ [5] and so our result is as follows:

Theorem 2.2. *Let X be a unary relation on $(\omega, \leq)$ that forms an infinite and coinfinite set. Suppose that there exists a computable copy $\boldsymbol{L}$ of $(\omega, \leq)$ such that $X_{\boldsymbol{L}}$ is computable. Then $DgSp_{(\omega,\leq)}(X) = \{\boldsymbol{d} : \boldsymbol{d} \leq \boldsymbol{0}'\}$.*

Before we prove Theorem 2.2, we need to recall some facts about Δ_2 sets, or equivalently, sets that are Turing-below $\emptyset'$. We call a function f *limit computable* if there exists a uniformly computable sequence of functions f_s such that $f = \lim_s f_s$. A set A is limit computable if its characteristic function χ_A is limit computable. The Limit Lemma states that a function f is limit computable if and only if $f \leq_T \emptyset'$[5]. We will use this fact to show that if X is order-computable, then the degree of $X_{\mathbf{L}}$ in any computable copy can be at worst Δ_2.

First we will show that every isomorphism between two computable copies of $(\omega, \leq)$ must be limit computable. We will let one of our copies be the standard copy for simplicity, but the same reasoning will work for two arbitrary copies. Suppose we have the standard copy $(\omega, \leq)$ and any arbitrary computable copy $\mathbf{L}$. Let φ be the isomorphism between the two orderings.

Since the order relation of $\mathbf{L}$, $\leq_{\mathbf{L}}$, is computable we can think of $\mathbf{L}$ as being revealed in a nice way. At stage 0, we set the domain of $\mathbf{L}$ equal to 0. At stage 1, we can compute whether $0 <_{\mathbf{L}} 1$ or $1 <_{\mathbf{L}} 0$ and place 1 in $\mathbf{L}$ accordingly. In general, at any stage $s+1$, with $\mathbf{L}[s]$ in hand, we can computably decide where $s+1$ belongs relative to the finitely many elements of $\mathbf{L}[s]$. We will use this process to show that the isomorphism, φ, between $(\omega, \leq)$ and $\mathbf{L}$ is limit computable.

$(\omega, \leq)$	0	1	2	3	4	5	6	7	8
$\mathbf{L}[0]$	0								

At stage 0 we have $|\mathbf{L}[0]| = \{0\}$ and thus we believe that $\varphi(0) = 0$. This will be the case until a number shows up $\leq_{\mathbf{L}} 0$. Suppose at stage 1 we compute that $0 <_{\mathbf{L}} 1$, and at stage 2 we determine that $2 <_{\mathbf{L}} 0 <_{\mathbf{L}} 1$.

$(\omega, \leq)$	0	1	2	3	4	5	6	7	8
$\mathbf{L}[1]$	0	1							

$(\omega, \leq)$	0	1	2	3	4	5	6	7	8
$\mathbf{L}[2]$	2	0	1						

Since 2 showed up to the left of 0 in $\mathbf{L}$, it now appears that $\varphi(0) = 2$. We will believe this is the case until a numbers shows up to the left of 2 in $\mathbf{L}$. Since $\mathbf{L} \cong (\omega, \leq)$, eventually numbers will stop appearing to the left of 0 in $\mathbf{L}$. The image of 0 in $\mathbf{L}$ at this stage is the true value of $\varphi(0)$.

By the same reasoning, φ will settle on each initial segment after a finite period of time, and hence we have a computable sequence φ_s such that $\varphi = \lim_s \varphi_s$. So φ is limit computable and hence $\emptyset'$-computable by the Limit Lemma.

Now we being the proof of Theorem 2.2. Since X is order-computable, there is a computable copy $\mathbf{L}$ for which $X_{\mathbf{L}}$ is computable. Let $\tilde{\mathbf{L}}$ be an arbitrary computable copy of $(\omega, \leq)$. Then we have $\varphi : \mathbf{L} \cong \tilde{\mathbf{L}}$ where $\varphi \leq_T \emptyset'$. Now let $x \in \omega$: we wish to determine whether $x \in X_{\tilde{\mathbf{L}}}$. From φ we can compute the preimage of x in $\mathbf{L}$, $\varphi^{-1}(x)$. Since $X_{\mathbf{L}}$ is computable, we can determine whether or not $\varphi^{-1}(x) \in X_{\mathbf{L}}$ and $x \in X_{\tilde{\mathbf{L}}} \Leftrightarrow \varphi^{-1}(x) \in X_{\mathbf{L}}$. We therefore have $X_{\tilde{\mathbf{L}}} \leq_T \varphi \leq_T \emptyset'$ and hence $DgSp_{(\omega,\leq)}(X) \subseteq \{\mathbf{d} : \mathbf{d} \leq \mathbf{0}'\}$ as desired. Now it remains to show that we can attain every Δ_2 degree by considering the image of X in an appropriate computable copy of $(\omega, \leq)$.

Consider the ordering $\mathbf{L} = (\omega, \leq_L) \cong (\omega, \leq)$ with $X_{\mathbf{L}}$ computable, and let Y be any Δ_2 set. Since Y is limit computable, there is a uniformly computable sequence of sets

$\{Y_s\}_{s\in\omega}$ such that $Y = \lim_s Y_s$. We call such a sequence a Δ_2-*approximating sequence* for Y. Our goal is to build another computable copy $\tilde{\mathbf{L}}$ of $(\omega, \leq)$ with $X_{\tilde{\mathbf{L}}} \equiv_T Y$.

By assumption, $\leq_{\mathbf{L}}$ is a computable relation so we can think of $\mathbf{L}$ as being revealed as previously discussed. Moreover, $X_{\mathbf{L}}$ is a computable set and so we can determine membership in $X_{\mathbf{L}}$ as soon as a number appears in $\mathbf{L}$. This time $(\mathbf{L}, X_{\mathbf{L}})$ is revealed as follows: At stage 0, we compute whether $0 \in X_{\mathbf{L}}$ or $0 \in \overline{X}_{\mathbf{L}}$. At stage 1, we compute whether $0 <_{\mathbf{L}} 1$ or $1 <_{\mathbf{L}} 0$, and determine whether or not $1 \in X_{\mathbf{L}}$. In general at stage $s+1$, we can computably decide where $s+1$ belongs relative to the finitely many elements of $\mathbf{L}[s]$ and determine whether or not $s+1$ is a member of $X_{\mathbf{L}}$.

While we reveal this ordering $\mathbf{L}$, we also want to build our own ordering $\tilde{\mathbf{L}}$ to match. In other words, we wish to build an isomorphism $\varphi : \omega \to \omega$ that matches each element in the domain of $\mathbf{L}$ with an appropriate element in the domain of our ordering $\tilde{\mathbf{L}}$. We will define φ on the elements of $|\mathbf{L}|$ in their ω-order (i.e. beginning with $0, 1, ...$) ensuring that, for each member x of the this domain, $\varphi(x) \in X_{\tilde{\mathbf{L}}} \Leftrightarrow x \in X_{\mathbf{L}}$.

At each stage, we will insert new numbers into our ordering $\tilde{\mathbf{L}}$ to maintain the partial isomorphism, while using the even numbers to code the set Y. (If x is even, then we will have $x \in X_{\tilde{\mathbf{L}}}$ at stage s if and only if $\frac{x}{2} \in Y_s$.) We will ensure that the set of odd numbers in $X_{\tilde{\mathbf{L}}}$ is computable by deciding whether or not an odd number is in $X_{\tilde{\mathbf{L}}}$ immediately when it is enumerated into the ordering $\tilde{\mathbf{L}}$, and we will ensure that the even numbers maintain their ω-order in $\tilde{\mathbf{L}}$ so that no number ends up with infinitely many predecessors.

We will first discuss a strategy for building an ordering $(\tilde{\mathbf{L}}, X_{\tilde{\mathbf{L}}})$ isomorphic to $(\mathbf{L}, X_{\mathbf{L}})$ while keeping the even numbers in $\overline{X}_{\tilde{\mathbf{L}}}$; we will later use the even numbers to code the set Y.

As mentioned above, we will first determine whether or not $0 \in X_{\mathbf{L}}$. Now we wish to introduce a possible match for 0 in $\tilde{\mathbf{L}}$ and the obvious match is, of course, 0. But if $0 \in X_{\mathbf{L}}$ then we wish to match 0 with a number that we can immediately put into $X_{\tilde{\mathbf{L}}}$. For this reason, if $0 \in X_{\mathbf{L}}$, then we enumerate the least odd number, 1, into $\tilde{\mathbf{L}}$ as $\varphi(0)$, and put 1 into $X_{\tilde{\mathbf{L}}}$ as desired. Now since we need to ensure that we enumerate all the even numbers eventually, we need to enumerate them whenever possible. It only

makes sense for us to enumerate an even number as a match for some number x in $\mathbf{L}$ if $x \notin X_{\mathbf{L}}$. Thus if $0 \notin X_{\mathbf{L}}$ then we will enumerate the least even number, 0, into $\tilde{\mathbf{L}}$ as the match for 0, and leave it out of our set $X_{\tilde{\mathbf{L}}}$. Thus we will have $\varphi(0) = 0$ or 1.

Now that an appropriate match has been determined for 0, we can continue. Since $\mathbf{L} \cong (\omega, \leq)$, we know that only finitely many numbers can appear $\leq_{\mathbf{L}}$-below 0. Thus anytime a new number x appears in $\mathbf{L}$ to the left of 0, we will just copy the least odd number available as its image in $\tilde{\mathbf{L}}$, and put the odd number into $X_{\tilde{\mathbf{L}}}$ if and only if $x \in X_{\mathbf{L}}$. This strategy will ensure that the segment $\leq_{\mathbf{L}}$-below 0 in $\mathbf{L}$ is indeed isomorphic to the segment $\leq_{\tilde{\mathbf{L}}}$-below $0_{\tilde{\mathbf{L}}}$ in $\tilde{\mathbf{L}}$. Now we must deal with what happens if $\mathbf{L}$ builds to the right of 0.

If a number x appears in $\mathbf{L}$ to the right of 0, then we will match x with an even or an odd number depending on its membership in $X_{\mathbf{L}}$. If $x \in X_{\mathbf{L}}$ then we will enumerate the least odd number not yet enumerated into $\tilde{\mathbf{L}}$ as $\varphi(x)$ and put $\varphi(x)$ into $X_{\tilde{\mathbf{L}}}$. If $x \notin X_{\mathbf{L}}$ then we have been given an opportunity to enumerate an even number as the image. We will enumerate the next even number into our ordering as $\varphi(x)$, keeping it out of our set $X_{\tilde{\mathbf{L}}}$.

Now we will give a more general outline of the method: When a number x appears in $\mathbf{L}$ with $X_{\mathbf{L}}(x) = 1$, then we define $\varphi(x)$ to be the least odd number not yet in $\tilde{\mathbf{L}}$ and put $\varphi(x)$ into $X_{\tilde{\mathbf{L}}}$. Let s_0 be the first stage when a number x appears in $\mathbf{L}$ with $X_{\mathbf{L}}(x) = 0$. At stage s_0 we will place the first even number, 0, in $\tilde{\mathbf{L}}$ as $\varphi(x)$. Now, let n be the largest even number in $\tilde{\mathbf{L}}[s]$ at any stage $s > s_0$, and let $m = \varphi^{-1}(n)$, the preimage of n in $\mathbf{L}$. If a number x appears $\leq_{\mathbf{L}}$-below m at stage s, then we copy an odd number as $\varphi(x)$ and put $\varphi(x)$ into $X_{\tilde{\mathbf{L}}}[s+1]$ if and only if $x \in X_{\mathbf{L}}$. If x appears to the right of m in $\mathbf{L}$, then we insert an odd number as a match if $x \in X_{\mathbf{L}}$ and an even number if $x \notin X_{\mathbf{L}}$. Since X is infinite and coinfinite by assumption, $X_{\mathbf{L}}$ is as well, so there are infinitely many bit alternations in $X_{\mathbf{L}}$. For this reason, we will always be able to find larger numbers both in and out of $X_{\mathbf{L}}$ and hence we will eventually enumerate all of ω into the domain of $\tilde{\mathbf{L}}$.

This strategy will certainly build $(\tilde{\mathbf{L}}, X_{\tilde{\mathbf{L}}}) \cong (\mathbf{L}, X_{\mathbf{L}})$, so now we must modify our method and use the even numbers to code our set Y and ensure that $X_{\tilde{\mathbf{L}}} \equiv_T Y$. In the end, we wish to have $2y \in X_{\tilde{\mathbf{L}}}$ if and only if $y \in Y$. If Y was c.e. then any time

a new element y appeared in $Y[s] - Y[s-1]$ we would enumerate $2y$ into $X_{\tilde{\mathbf{L}}}$ and be done with that coding. Since we are only asking that Y be Δ_2, the set Y can remove y at a later stage, leaving us no choice but to change our minds about the membership of $2y$ in $X_{\tilde{\mathbf{L}}}$. We will choose the Δ_2-approximating sequence, $\{Y_s\}_{s\in\omega}$, for Y such that $Y_s \subseteq \{1, 2, ..., s-1\}$ and Y_{s+1} differs from Y_s by at most one number. Using this sequence we will ensure that, at each stage s of our construction, $2y \in X_{\tilde{\mathbf{L}}}[s]$ if and only if $y \in Y_s$. We will now examine how coding Y will affect the construction of our isomorphism φ.

Let s be the first stage where $Y_s \neq \emptyset$ and say $\{y_0\} = Y_s - Y_{s-1}$. To code Y, we will put $2y_0$ into our set $X_{\tilde{\mathbf{L}}}$ as desired. Note that by our discussion above, $2y_0$ must be in $\overline{X}_{\tilde{\mathbf{L}}}$ at stage s, since all even numbers start out of $X_{\tilde{\mathbf{L}}}$ until Y exerts its power. Since we must change our minds about the membership of $2y_0$, we have now damaged the partial isomorphism that we had been building as described in the previous paragraphs. In particular, we have $2y_0 \in X_{\tilde{\mathbf{L}}}[s]$ while $\varphi^{-1}(2y_0) \notin X_{\mathbf{L}}[s]$.

Let us consider a particular example of $\mathbf{L}$. As $\mathbf{L}$ reveals itself, we will build $\tilde{\mathbf{L}}$ to match, following our method. In the example, we will use boxes to indicate membership in the sets $X_{\mathbf{L}}$ and $X_{\tilde{\mathbf{L}}}$.

Stage 0 :	$\mathbf{L}$	**0**				**0**		
	$\tilde{\mathbf{L}}$				$\longrightarrow$	**0**		
Stage 1 :	$\mathbf{L}$	0	**1**			0	**1**	
	$\tilde{\mathbf{L}}$	0			$\longrightarrow$	0	**2**	
Stage 2 :	$\mathbf{L}$	0	**2**	1		0	**2**	1
	$\tilde{\mathbf{L}}$	0		2	$\longrightarrow$	0	**1**	2

We can follow our matching scheme until the first number appears in Y. Suppose that Y enumerates 1 as its first member at stage 5.

Stage 3 :	$\mathbf{L}$		0	2	1	**3**			0	2	1	**3**
	$\tilde{\mathbf{L}}$		0	1	2		$\longrightarrow$		0	1	2	**4**
Stage 4 :	$\mathbf{L}$	$\boxed{\mathbf{4}}$	0	2	1	3		$\boxed{\mathbf{4}}$	0	2	1	3
	$\tilde{\mathbf{L}}$		0	1	2	4	$\longrightarrow$	$\boxed{\mathbf{3}}$	0	1	2	4
Stage 5 :	$\mathbf{L}$	$\boxed{4}$	0	2	1	3		$\boxed{4}$	0	2	1	3
	$\tilde{\mathbf{L}}$	$\boxed{3}$	0	1	2	4	$\longrightarrow$	$\boxed{3}$	0	1	$\boxed{\mathbf{2}}$	4

Since $1 \in Y_5 - Y_4$, we must put 2 into $X_{\tilde{\mathbf{L}}}$ at stage 5. It is now clear that our choice for $\varphi(1)$ is no longer valid and hence we must change our isomorphism, but we wish to maintain as much of the isomorphism as we can. Do do so, we can insert a new odd number into $\tilde{\mathbf{L}}$ as a new match for 1, leaving the odd number out of $X_{\tilde{\mathbf{L}}}$ since $1 \notin X_{\mathbf{L}}$. In our example, suppose we insert the next odd number, 5, as follows:

Stage 5 :	$\mathbf{L}$	$\boxed{4}$	0	2	1	3		$\boxed{4}$	0	2	1	3	
	$\tilde{\mathbf{L}}$	$\boxed{3}$	0	1	$\boxed{2}$	4		$\boxed{3}$	0	1	***5***	$\boxed{2}$	4

This will leave the values of $\varphi(n)$ intact for all $n <_{\mathbf{L}} 1$. While 1 now has an appropriate match in $\tilde{\mathbf{L}}$, it is clear that we have not absorbed the problem completely. In particular, 3 is now unhappy sitting above a number in $X_{\tilde{\mathbf{L}}}$. It is clear that inserting one new number will not rectify the problem so we now will try a more sophisticated approach. We will now return to the general case, and to reflect the fact that we will be making alterations to our partial isomorphism as we go, we will denote the isomorphism we have at stage s as φ_s. Also, we will denote the segment $\{l : l \geq_{\mathbf{L}} x\}$ of $\mathbf{L}$ as $\mathbf{L}_{\geq x}$ and the segment $\{l : l >_{\mathbf{L}} x\}$ as $\mathbf{L}_{>x}$.

Let us suppose that y_0 is enumerated into Y at stage s. Then we put $2y_0$ into $X_{\tilde{\mathbf{L}}}[s]$ as required. Let $z = \varphi_s^{-1}(2y_0)$. We will maintain the partial isomorphism φ_s on the segment $\mathbf{L}\!\upharpoonright\! z$ and speed up the enumeration of $\mathbf{L}$ until we can "absorb" the segment

$\tilde{\mathbf{L}}_{\geq 2y_0}$ into $\mathbf{L}_{\geq z}$. $\tilde{\mathbf{L}}_{\geq 2y_0}$ is a finite segment with a certain configuration of 0's and 1's, depending on membership in $X_{\tilde{\mathbf{L}}}$. To match this configuration in $\mathbf{L}$, we must speed up the enumeration of $\mathbf{L}$ until we can embed our configuration into $\mathbf{L}_{\geq z}$. (For example, a sequence of 101 appearing in $\tilde{\mathbf{L}}$ can be embedded into any configuration of the form ...101..., ...1001..., etc.) Since $X_{\mathbf{L}}$ is infinite and coinfinite and each number in $\mathbf{L}$ has finitely many predecessors, eventually we must see a desired configuration. Going back to our example:

$\mathbf{L}$	$\boxed{4}$	0	2	|	1	3	
$\tilde{\mathbf{L}}$	$\boxed{3}$	0	1	|	$\boxed{2}$	4	

Let us suppose that after speeding up the enumeration of $\mathbf{L}$ we get the following configuration in $\mathbf{L}$:

$\mathbf{L}$	$\boxed{4}$	0	**5**	2	|	**6**	1	3	$\boxed{\mathbf{7}}$	**8**
$\tilde{\mathbf{L}}$	$\boxed{3}$	0		1	|		$\boxed{2}$	4		

It is now possible to embed $\tilde{\mathbf{L}}_{>1}$ into $\mathbf{L}_{>2}$. In this case, an (order-preserving) embedding β is given by: $\beta(2) = 7$, $\beta(4) = 8$. We will now define the new isomorphism φ_s on the larger domain of $\mathbf{L}$ ensuring that if $n = \beta(m)$ for some m, then $\varphi_s(n) = m$. In our case, we must have $\varphi_s(7) = 2$ and $\varphi_s(8) = 4$. We make φ_s a bijection by appropriately inserting new even and odd numbers into $\tilde{\mathbf{L}}$.

$\mathbf{L}$	$\boxed{4}$	0	**5**	2	|	**6**	1	3	$\boxed{\mathbf{7}}$	**8**
$\tilde{\mathbf{L}}$	$\boxed{3}$	0	**5**	1	|	**7**	**9**	**11**	$\boxed{2}$	4

By the end of stage s, we will have an order-preserving bijection, φ_s, defined between $\mathbf{L}[s]$ and $\tilde{\mathbf{L}}[s]$, and we will have $x \in X_{\mathbf{L}}[s] \Leftrightarrow \varphi_s(x) \in X_{\tilde{\mathbf{L}}}$.

We treat the case where Y removes a number at stage s (eg. $\{y_0\} = Y_{s-1} - Y_s$), in a similar manner. If this occurs, then our coding strategy tells us to remove $2y_0$ from $X_{\tilde{\mathbf{L}}}$ at stage s. This will of course damage the isomorphism we had built by the end of stage $s-1$ since, in particular, $2y_0 \notin X_{\tilde{\mathbf{L}}}[s]$ while $z = \varphi_{s-1}^{-1}(2y_0) \in X_{\mathbf{L}}[s]$. Again, we will consider the finite sequence appearing in $\tilde{\mathbf{L}}_{\geq 2y_0}$ and speed up the enumeration of $\mathbf{L}$ in order to embed this configuration into $\mathbf{L}$ by the end of stage s.

Since $X_{\mathbf{L}}$ has infinitely many bit alternations, we will be able to absorb any finite sequence we find in $\tilde{\mathbf{L}}$ be redefining φ on finitely many values. At the end of each stage, we will indeed have a proper isomorphism between the finite orderings as desired. Moreover, we will argue that in the limit we will have a true isomorphim $\varphi = \lim_s \varphi_s$ between $\mathbf{L}$ and $\tilde{\mathbf{L}}$. We will do so by ensuring that we change the isomorphism only finitely often on each value of the domain and range.

Construction

Fix a Δ_2-approximation of Y with $\lim_s Y_s = Y$ such that $Y_s \subseteq \{1, 2, ..., s-1\}$ and Y_{s+1} differs from Y_s by at most one number. At stage 0, determine whether or not $0 \in X_{\mathbf{L}}$, choose the appropriate value for $\varphi_0(0)$ as described above and enumerate $\varphi_0(0)$ into $\tilde{\mathbf{L}}$.

At stage $s+1$:

Step 1: Suppose that one of $Y_{s+1} - Y_s$ or $Y_s - Y_{s+1}$ is non-empty. If $y \in Y_{s+1} - Y_s$ then put $2y$ into $X_{\tilde{\mathbf{L}}}[s+1]$. If $y \in Y_s - Y_{s+1}$ then put $2y$ into $\overline{X}_{\tilde{\mathbf{L}}}[s+1]$.

Step 2: If we acted in Step 1, then our partial isomporphism φ_s is no longer valid on the segment $\geq_{\mathbf{L}} \varphi_s^{-1}(2y)$ in $\mathbf{L}$. In particular, $X_{\mathbf{L}}(\varphi_s^{-1}(2y))[s+1] \neq X_{\tilde{\mathbf{L}}}(2y)[s+1]$. We wish to preserve as much of φ_s as possible so we will build φ_{s+1} such that φ_{s+1} agrees with φ_s on the initial segment of $\mathbf{L}$, $\mathbf{L}\restriction\varphi_s^{-1}(2y)$. We will build φ_{s+1} as follows:

Let $\varphi_{s+1}(n) = \varphi_s(n)$ for all $n <_{\mathbf{L}} \varphi_s^{-1}(2y)$. Speed up the enumeration of $\mathbf{L}$ until we have an (order-preserving) embedding, call it β, of $\tilde{\mathbf{L}}_{>2y}$ into $\mathbf{L}_{>\varphi_s^{-1}(2y)}$. Extend β to a bijective mapping φ_{s+1} by inserting numbers into $\tilde{\mathbf{L}}$ as needed, working from left to right. If $z = \beta(x)$ for some $x \in \tilde{\mathbf{L}}$ then let $\varphi_{s+1}(z) = x$. If z has no match in $\tilde{\mathbf{L}}$ (i.e. z is not in the image of our chosen embedding β) then we need to introduce a new even or odd number as $\varphi_{s+1}(z)$. If the largest even number in $\tilde{\mathbf{L}}$ is matched to a number $\leq_{\mathbf{L}}$-above z, then insert the largest odd number not yet in $\tilde{\mathbf{L}}$ as $\varphi_{s+1}(z)$ and put $\varphi_{s+1}(z)$ into $X_{\tilde{\mathbf{L}}}$ if and only if $z \in X_{\mathbf{L}}$. If the largest even number is matched to a number

$\leq_{\mathbf{L}}$-below z then insert an odd number as $\varphi_{s+1}(z)$ if $z \in X_{\mathbf{L}}$ and an even number as $\varphi_{s+1}(z)$ if $z \notin X_{\mathbf{L}}$. Note that, by the end of Step 2, we have $\varphi_{s+1} : \mathbf{L}[s+1] \cong \tilde{\mathbf{L}}[s+1]$.

Step 3: Finally, we will ensure that $2s$ and $2s+1$ have been placed in the $\tilde{\mathbf{L}}$-ordering by the end of stage $s+1$. If this occured in Step 2, then we are done. If not, we will speed up the enumeration of $\mathbf{L}$ as in Step 2 until we find an appropriate place to insert $2s$ and $2s+1$ into $\tilde{\mathbf{L}}$.

End Construction

Verification

It is not hard to see that $\tilde{\mathbf{L}}$ is indeed a computable ordering. To compute whether $a \leq_{\tilde{\mathbf{L}}} b$ we run the construction until we see the first stage s where both a and b are in $\tilde{\mathbf{L}}[s]$. Such a stage must occur since the domain of $\tilde{\mathbf{L}}$ is all of ω. At stage s we can determine whether or not $a \leq_{\tilde{\mathbf{L}}[s]} b$. By our construction, the position of a relative to b in $\tilde{\mathbf{L}}$ will not change once they appear together in $\tilde{\mathbf{L}}[s]$, and hence $a \leq_{\tilde{\mathbf{L}}} b$ if and only if $a \leq_{\tilde{\mathbf{L}}[s]} b$.

Now it remains to show that $\tilde{\mathbf{L}} \cong (\omega, \leq)$ and $(\omega, \leq_{\tilde{\mathbf{L}}}, X_{\tilde{\mathbf{L}}}) \cong (\omega, \leq_{\mathbf{L}}, X_{\mathbf{L}})$ with $X_{\tilde{\mathbf{L}}} \equiv_T Y$.

Lemma 2.2.1. *The isomorphism changes finitely often on each element of its domain and range.*

Proof. First we will consider the domain $\mathbf{L}$. Let $x \in \mathbf{L}$ and let t be a stage where $\mathbf{L}_t{\restriction}x = \mathbf{L}{\restriction}x$. I.e. A stage when $\mathbf{L}$ has finished enumerating numbers $\leq_{\mathbf{L}}$-below x. Such a stage must exist since $\mathbf{L} \cong (\omega, \leq)$. Let n be the largest even number $\leq_{\tilde{\mathbf{L}}}$-below $\varphi_t(x)$ in $\tilde{\mathbf{L}}$ at stage t. We now claim that every even number that is $\leq_{\tilde{\mathbf{L}}}$-less than $\varphi_s(x)$, at any stage $s > t$, is less than or equal to n.

The only way that the value of $\varphi(x)$ changes is if coding an even number, call it k, below $\varphi_t(x)$ destroys our isomorphism somewhere in $\tilde{\mathbf{L}}{\restriction}\varphi_t(x)$. Suppose this occurs at stage $s > t$. We will then speed up the enumeration of $\mathbf{L}$ to find an embedding β to absorb this induced problem. We claim that $\varphi_{s+1}(x) \leq_{\tilde{\mathbf{L}}} \varphi_s(x)$. We will first illustrate

this with an example. Suppose that we have the following isomorphism at stage $s-1$ and that 0 enters Y at stage s:

Stage s :	$\mathbf{L}$	2	6	$\boxed{0}$	3	|	1	4	$\boxed{5}$	7
	$\tilde{\mathbf{L}}$	3	9	$\boxed{1}$	5	|	$\boxed{\mathbf{0}}$	2	$\boxed{7}$	4

Let us assume that $\mathbf{L}$ is done enumerating below 7 at stage s. Then we must have $\beta(0) \geq_{\mathbf{L}} 5$ for our embedding β to be order preserving and so our extending isomorphism φ_{s+1} must satisfy $\varphi_{s+1}(5) \leq_{\tilde{\mathbf{L}}} 0$. Since φ_{s+1} preserves order, we must have $\varphi_{s+1}(7) \leq_{\tilde{\mathbf{L}}} 2 < \varphi_s(7)$.

In general, since $\mathbf{L}$ can no longer enumerate below x we must have $\beta(k) > \varphi_s^{-1}(k)$ and thus $\beta(\varphi_s(x)) > x$. By how we extend β in the construction, we have $\varphi_{s+1}(x) > \varphi_s(x)$ as desired. By this claim, we know that the set of even numbers lying $\leq_{\tilde{\mathbf{L}}}$-below $\varphi_{t'}(x)$ at any stage $t' > t$ are in the set $\{0, 2, ..., n\}$.

Let $t' > t$ be a the stage when $Y_{t'} \upharpoonright \frac{n}{2} = Y \upharpoonright \frac{n}{2}$, then after stage t', Y will not change the membership of any even number lying to the left of $\varphi_{t'}(x)$ in the ordering $\tilde{\mathbf{L}}$. Hence, by our construction, the value of $\varphi(x)$ will remain unchanged after stage t'.

Now we will consider the elements in the range of φ. Suppose that $x \in \tilde{\mathbf{L}}$. We will consider the case where x is even and odd separately. Suppose that x is even. Let t be a stage where $Y_t \upharpoonright \frac{x}{2} = Y \upharpoonright \frac{x}{2}$. Since, by construction, all even numbers appear in $\tilde{\mathbf{L}}$ in their natural order, we know that no even number $\leq_{\tilde{\mathbf{L}}} x$ can be enumerated into $X_{\tilde{\mathbf{L}}}$ after stage t. By the end of stage t we will have built a partial isomorphism between $\mathbf{L}[t]$ and $\tilde{\mathbf{L}}[t]$, and we know that Y cannot destroy the matching between the initial segments $\mathbf{L} \upharpoonright \varphi_t^{-1}(x)$ and $\tilde{\mathbf{L}} \upharpoonright x$ after stage t. Thus $\varphi_{t'}^{-1}(x) = \varphi_t^{-1}(x)$ for all $t' \geq t$.

Now suppose that x is odd. Let t_0 be the first stage where an even number, say $z+2$, appears $\leq_{\tilde{\mathbf{L}}}$-above x in $\tilde{\mathbf{L}}$. We thus have $z \leq_{\tilde{\mathbf{L}}} x \leq_{\tilde{\mathbf{L}}} z+2$, and by construction, no other even number can appear $\leq_{\tilde{\mathbf{L}}}$-below $z+2$ after stage t_0. Let $t_1 > t_0$ be a stage when $Y_{t_1} \upharpoonright \frac{z}{2} = Y \upharpoonright \frac{z}{2}$. By our construction, Y cannot change the membership of any even number lying $\leq_{\tilde{\mathbf{L}}}$-below x after stage t_1 and thus $\varphi_{t_1}^{-1}(x)$ is the true preimage of x in $\mathbf{L}$. Hence the isomorphism changes finitely often on each element in the range, $\tilde{\mathbf{L}}$. $\square$

Lemma 2.2.2. *Every element in the domain of $\tilde{\boldsymbol{L}}$ has finitely many predecessors*

Proof. By Lemma 2.2.1, there is a stage s where the true preimage of x in $\mathbf{L}$ has been determined. Let $z = \varphi_s^{-1}(x)$, then $x = z_{\tilde{\mathbf{L}}}$. Our number x will gain a predecessor after stage s if and only if a number appears $\leq_{\mathbf{L}}$-below z in $\mathbf{L}$. Since $\mathbf{L} \cong (\omega, \leq)$, this can occur only finitely often, and hence x has finitely many predecessors in $\tilde{\mathbf{L}}$. □

Lemma 2.2.3. $X_{\tilde{\boldsymbol{L}}} \equiv_T Y$.

Proof. We have $X_{\tilde{\mathbf{L}}} = \lim_s X_{\tilde{\mathbf{L}}}[s]$, and by our construction, $y \in Y \Leftrightarrow 2y \in X_{\tilde{\mathbf{L}}}$ and therefore $Y \leq_T X_{\tilde{\mathbf{L}}}$.

Now let $x \in \omega$. If x is odd, then we can compute whether or not $x \in X_{\tilde{\mathbf{L}}}$ by waiting for the stage s where x is enumerated into the ordering $\tilde{\mathbf{L}}$. At this stage, we will either put x into $X_{\tilde{\mathbf{L}}}[s]$ or declare $x \in \overline{X}_{\tilde{\mathbf{L}}}$ and, by our construction, this will remain unchanged. If x is even, then $x \in X_{\tilde{\mathbf{L}}} \Leftrightarrow \frac{x}{2} \in Y$. So $X_{\tilde{\mathbf{L}}} \leq_T Y$, and hence $X_{\tilde{\mathbf{L}}} \equiv_T Y$ as desired. □

By Lemma 2.2.1 and 2.2.2 we have that $(\omega, \leq_{\tilde{\mathbf{L}}}, X_{\tilde{\mathbf{L}}}) \cong (\omega, \leq_{\mathbf{L}}, X_{\mathbf{L}})$ via the isomorpism $\varphi = \lim_s \varphi_s$. By Lemma 2.2.3, we have $deg(X_{\tilde{\mathbf{L}}}) = deg(Y)$ as desired.

We have shown that we can construct a computable copy $\tilde{\mathbf{L}}$ of $(\omega, \leq)$ where $X_{\tilde{\mathbf{L}}}$ has any given Δ_2 degree. Hence, if the degree spectra of an infinite and coinfinite set X contains the computable degree, then $DgSp_{(\omega,\leq)}(X)$ contains all Δ_2 degrees. Theorem 2.2 gives us a nice description of the degree spectrum of order-computable sets. The next question one can ask is what kinds of sets have this property.

Suppose that X is a unary relation on $(\omega, \leq)$ that forms a c.e. set on the standard copy. In [3] the authors study the question of determining what conditions on a set X will ensure that the degree spectrum of X contains the computable degree. A result from [3] shows that if the degree of a c.e. set X is low (ie. $deg(X)' = \mathbf{0}'$), then we indeed have $\mathbf{0} \in DgSp_{(\omega,\leq)}(X)$. This proof is outlined in [3] and relies on a characterization of low c.e. sets. It turns out that it is possible to do better. It is not necessary for the set X to have low degree, it is actually enough that the c.e. set X not be of high degree. I.e. $X' <_T \emptyset''$.

Theorem 2.3 (Downey, Khoussainov, Miller and Yu). *Let X be a unary relation on $(\omega, \leq)$. If X forms a c.e. set whose Turing degree is not high, then $\boldsymbol{0} \in DgSp_{(\omega,\leq)}(X)$.*

The proof of this theorem is presented in full in [2] and uses Martin's Characterization of high degrees.

2.3 Degree Spectra of Sets of High c.e. Degree

Now we will move on to a result from [2] showing that Theorem 2.3 is as good as we can do for producing categories of order-computable sets. It turns out that each high c.e. degree contains a set that is not order-computable.

Theorem 2.4 (Downey, Khoussainov, Miller and Yu). *Let $\boldsymbol{d}$ be a high c.e. Turing degree. Then there exists a c.e unary relation X on $(\omega, \leq)$ such that $deg(X) = \boldsymbol{d}$ and X is not order computable.*

This result is presented in [2], although a full construction and verification was not provided. We will provide it here.

To prove this theorem we will fix a c.e. set A in our given high c.e. degree and build a set X, Turing-equivalent to A, that is not order computable. In other words we must construct $X \subset \omega$ such that if $\mathbf{L}$ is a computable copy of $(\omega, \leq)$, then $X_{\mathbf{L}}$ is not computable. First we need to produce an effective list of all possible countable computable linear orderings, and to do so, we will use the fact that every such ordering is a subordering of the rational numbers. We will drop the word "countable" from now on since, in this paper, we are only concerned with countable domains. We will show that each c.e set produces a computable linear ordering, and in turn, each such ordering can be represented as a c.e. set.

Let q_0, q_1, q_2,... be an effective list of the rational numbers. Let L be a c.e. set with enumeration $\{L_s\}_{s\in\omega}$. We define our linear ordering $\mathbf{L} = (A, \leq_{\mathbf{L}})$ as follows: Suppose that L enumerates $i \in \omega$ at stage $s+1$ and let $L_s = \{i_1, ..., i_k\}$. We put the least $n \in \omega$

that is not yet in $\mathbf{L}$ into $\mathbf{L}$ and declare $n \leq_{\mathbf{L}} i_j$ if and only if $q_i \leq_{\mathbb{Q}} q_{i_j}$ for $j = 1, ..., k$. To compute whether $n_1 \leq_{\mathbf{L}} n_2$ we simply have to wait for the first stage s where n_1 and n_2 appear together in $\mathbf{L}[s]$ and $n_1 \leq_{\mathbf{L}} n_2$ if and only if $n_1 \leq_{\mathbf{L}[s]} n_2$.

Note that since, in the above construction, we are enumerating the natural numbers into $\mathbf{L}$ in their ω-order, the domain of $\mathbf{L}$ will either be ω or a finite initial segment segment of ω. Of course, the c.e. set L can "correspond" to a computable ordering that is not isomorphic to $(\omega, \leq)$. For example, L could be finite and hence $\mathbf{L}$ will have a finite initial segment of ω as its domain. Or L could enumerate numbers in such a way that the induced ordering $\mathbf{L}$ has no endpoints, or is dense.

Now let $\mathbf{L}$ be a computable linear ordering with domain $\mathbb{N}$. We will construct our c.e. set L as follows: First enumerate 0 into L. Next compute whether $0 <_{\mathbf{L}} 1$ or $1 <_{\mathbf{L}} 0$. If we have the former case, find the least i such that $q_0 <_{\mathbb{Q}} q_i$; in the latter case, find the least i such that $q_i <_{\mathbb{Q}} q_0$. In either case, enumerate i into L. In general, let $L_s = \{i_0, ..., i_s\}$. At stage $s+1$ we determine the position of $s+1$ in $\mathbf{L}$ relative to the numbers $< s+1$. Find the smallest index i_{s+1} such that, for all $j \leq s$, $q_{i_{s+1}} <_{\mathbb{Q}} q_{i_j}$ if and only if $s+1 <_{\mathbf{L}} j$. Since the rationals are dense, such a number $q_{i_{s+1}}$ must exist. Finally, enumerate i_{s+1} into L_{s+1}.

So every computable linear ordering can be represented as a c.e. set. Let $\{L_e : e \in \omega\}$ be a uniformly effective list of all c.e. sets, then this will effectively list all possible computable linear orderings. By our discussion above, this effective list will contain all computable copies of $(\omega, \leq)$, along with computable linear orderings of other order types.

Let us suppose that we have a linear ordering $\mathbf{L}$ (corresponding to a c.e. set L) and let U be a unary relation on the domain of $\mathbf{L}$. Then U determines a subset of the natural numbers. Moreover, the set U is computable if and only if U and $\overline{U}$ are (disjoint) c.e. sets. We can also produce a uniformly effective list of all pairs of c.e. sets, say $\{(W_e, V_e) : e \in \omega\}$, and thus U is computable if and only if the pair $(U, \overline{U})$ appears in this list. So a unary relation X on $(\omega, \leq)$ is order-computable only if we there exists a triple, $\langle L, W, V \rangle$ with the following properties:

1. L, W and V are c.e. sets

2. $W \cap V = \emptyset$

3. $\mathbf{L} \cong (\omega, \leq)$ ($\mathbf{L}$ being the linear ordering induced by L)

4. $(X_{\mathbf{L}}, \overline{X}_{\mathbf{L}}) = (W, V)$

We can think of the proof of this theorem as a game between us and an opponent that is trying to build such a triple. The opponent's goal is to build a triple, $\langle L, W, V \rangle$, satisfying properties $1 - 4$, and hence a triple witnessing the fact that X is order-computable, and our goal is to build our set $X \equiv_T A$ such that no such triple exists. If we develop a strategy to build X in such a way that the opponent fails, then we win and we have proven the theorem.

During the construction, the opponent is building the sets L, W and V in stages, ensuring that the sets are c.e. (i.e. $L = \cup_s L_s$, $W = \cup_s W_s$ and $V = \cup_s V_s$). L will be producing a computable linear ordering $\mathbf{L}$ in stages and W and V will be guessing at $X_{\mathbf{L}}$ and $\overline{X}_{\mathbf{L}}$ respectively. Since W must list $X_{\mathbf{L}}$, each time W enumerates a new number, y, the triple has declared that it believes y is in the set $X_{\mathbf{L}}$. Similarly, since V must list $\overline{X}_L$, each time V enumerates a number, that number enters the triple's version of $\overline{X}_{\mathbf{L}}$.

At each stage s, the opponent is trying to produce a partial isomorphism between our ordering, $(\omega, \leq, X)[s]$, and his ordering, $\langle L, W, V \rangle [s]$, by enumerating new numbers into the sets L, W and V at stage s. (Note that he cannot remove any numbers since these sets must be c.e.) To counter this, whenever we see a partial isomorphism between some initial segment of $(\omega, \leq, X)$ and $\langle L, W, V \rangle$ at stage s, we will destroy the matching by enumerating a new number into X within that initial segment. (Again, since X must be c.e., we cannot remove any numbers from X.)

We need to ensure that the opponent will fail no matter how he chooses to enumerate into L, W and V. Luckily, we can list all possible triples effectively, and so we have a good way of coding all possible strategies of our opponent. Let $\{\langle L_e, W_e, V_e \rangle : e \in \omega\}$ form a uniformly effective list containing all possible combinations of triples where L_e,

W_e, V_e are c.e sets and $W_e \cap V_e = \emptyset$. To succeed, our construction needs to meet the following requirements for each $e \in \omega$:

$$Q_e : \mathbf{L}_e \cong (\omega, \leq) \Longrightarrow X_{\mathbf{L}_e} \neq W_e \text{ or } \overline{X}_{L_e} \neq V_e$$

If our construction meets Q_e for all $e \in \omega$ then we win; if there is some $e \in \omega$ such that Q_e is not met, then our opponent wins. Our goal is to outline a construction that has only two possible outcomes for the given triple: either W_e and V_e do not witness X_{L_e} being computable, or L_e eventually builds an ordering that is not isomorphic to $(\omega, \leq)$. In either case, Q_e is met.

Note that from now on, we will not differentiate between L_e, the c.e. set, and $\mathbf{L}_e$, the induced linear ordering. We will simply write L_e to reference either one since the correspondence should be clear from our previous discussions.

We will now outline a construction to meet one requirement Q_e. In our construction, we need not act until it appears that Q_e is a "threat", or, in other words, until it appears that Q_e may not be met. First we need to define what we mean by a requirement being a "threat".

At the begining of our construction, we will produce a list of numbers $\{x^i : i \in \omega\} \subset \omega$ that we will enumerate into X. This countable list of natural numbers will divide $(\omega, \leq)$ into intervals. In our construction, we will wait for lengths of agreement on the intervals $[0, x^i]$ for each i in turn. Before we proceed, we must decide what we mean by a length of agreement on the entire interval $[0, x^i]$. We will wait until we see a stage s when $L_{e,s}$ contains at least $x^i + 1$ elements, and each of these $x^i + 1$ numbers has been enumerated by either W_e or V_e by stage s. At this moment, the pair $(W_e, V_e)[s]$ has made a guess at whether each of the first $x^i + 1$ numbers of $(\omega, \leq)$ are in or out of X. For each $x \in \omega$, let x_{L_e} be the image of x in L_e. We say we have a length of agreement up to x^i at this stage s if for each $x \in X_s \upharpoonright x^i$,

$$x \in X_s \Leftrightarrow x_{L_e} \in W_{e,s} \text{ and } x \notin X_s \Leftrightarrow x_{L_e} \in V_{e,s}.$$

Using the terminology from [2], we say that x^i is "realized" for Q_e at stage s if $\langle L_e, W_e, V_e \rangle [s]$ has a length of agreement up to x^i at this stage.

We will now outline our strategy for enumerating into X: First, we will do nothing until we see an agreement between $\langle L_e, W_e, V_e\rangle\,[s]$ and $(\omega, \leq, X_s)$ on an entire interval $[0, x^i]$. If this occurs, then we will enumerate a number in the interval $[0, x^i)$ to destroy this length of agreement. Now the isomorphism that the opponent had built is no longer valid, and thus Q_e is no longer a threat for the time being. After stage s, there are two possibilities: Either we never again see an agreement between $\langle L_e, W_e, V_e\rangle\,[t]$ and $(\omega, \leq, X)[t]$ on $[0, x^i]$ for any stage $t > s$, or there is a stage $t > s$ such that we see x^i realized once again at stage t. If we have the former case, then $\langle L_e, W_e, V_e\rangle$ will not be the winning triple for the opponent; if we have the latter case, then the sets L_e, W_e and V_e have enumerated in such a way that Q_e is again a threat to us.

We wish to control how easily the triple $\langle L_e, W_e, V_e\rangle$ can recover and impose new agreements. We will do so by enumerating very specific numbers in our intervals $[0, x^i]$ as follows:

Let's suppose that at stage s, x^0 is realized for Q_e. Since at stage 0 the set X contains only the x^i's and we only enumerate into X after x^0 is realized, $(0, x^0 - 1) \subset \overline{X_s}$. So in order to have an agreement up to x^0 we must have $y_0 \in W_{e,s}$ with $\left|\{z : z <_{L_{e,s}} y_0\}\right| = x^0$ and $\{z : z <_{L_{e,s}} y_0\} \subset V_{e,s}$. Hence y_0 is the apparent image of x^0 in L_e. At this stage, we wish to derail the isomorphism between $\langle L_e, W_e, V_e\rangle\,[s]$ and $(\omega, \leq, X_s)$ up to x^0. We will do so by enumerating $x^0 - 1$ into X. Since $\{z : z <_{L_{e,s}} y_0\} \subset V_{e,s}$, the opponent cannot enumerate any $z <_{L_{e,s}} y_0$ into X_{L_e} as the potential image of $x^0 - 1$. The opponent must then enumerate at least one new rational number $<_{L_e}$-below y_0 to maintain the partial isomorphism. This new member of L_e will ensure that y_0 is now the apparent image of some natural number strictly to the right of x^0 in $(\omega, \leq)$. At this point, we wait and see what our opponent does, and we extend our area of interest to the larger interval $[0, x^1]$. We will not act again until L_e recovers on this larger interval and we see a stage $s' > s$ where x^1 is realized for Q_e.

At stage s', by our argument above, y_0 must be to the right of $x^0_{L_{e,s'}}$. In fact, since $[x^0 + 1, x^1 - 1] \subset \overline{X}_{s'}$, the next possible matching for y_0 is x^1. We can thus conclude that if such a stage s' occurs, then we have $y_0 \geq x^1_{L_{e,s'}}$. Now that we have an agreement on the entire interval $[0, x^1]$ we wish to repeat our strategy and this time, we enumerate the predecessor of x^1 to derail the apparent matching. By similar reasoning, the opponent

has no choice but to enumerate a new element $<_{L_e}$-below y_0 as a possible image for $x^1 - 1$, thus shifting y_0 to the right of $x^1_{L_e}$. Now we wait for L_e to recover on the even larger interval $[0, x^2]$ and argue that if there exists a stage s'' where x^2 is realized for Q_e, then at this stage $y_0 \geq x^2_{L_{e,s''}}$. In general, we can argue that if there is a stage t where x^i is realized, then y_0 must be strictly to the right of $x^{i-1}_{L_{e,t}}$ in L_e.

Now, we focus our attention on this y_0. Let us suppose that after some stage, no new x^i is realized. Then from some point on, L_e can no longer be be considered isomorphic to $(\omega, \leq, X)$ as computed by (W_e, V_e). In this case, Q_e is satisfied since we have no isomorphism. If x^i is realized for all $i \in \omega$ then y_0 will be strictly to the right of the apparent image of each x^{i-1} in L_e. Thus L_e has an element with infinitely many predecessors, and hence cannot be isomorphic to $(\omega, \leq)$. In either case, Q_e is satisfied. This strategy will produce a set $X \subseteq \omega$ that is not witnessed to be order-computable by $\langle L_e, W_e, V_e \rangle [s]$, and all that we require is enough room between the x^i's so that we can enumerate whenever necessary. At this point, it appears that we are required to enumerate just once under each x^i to shift y_0 as desired, so one number in between each of the x^i's will suffice. It turns out that we will need more room due to the second condition in our theorem: we require that $X \equiv_T A$, the fixed c.e. set in our given high c.e. degree. To ensure that A is computable from X, we will use our master list $\{x^i : i \in \omega\} \subset \omega$ for coding, and to ensure that $X \leq_T A$, we will use the method of high permitting.

Let $\bar{A} = \{a_0 < a_1 < a_2 < \ldots\}$ and $\bar{A}_s = \{a^s_0 < a^s_1 < a^s_2 < \ldots\}$. Then the computation function of $\bar{A}$ is defined as $c_{\bar{A}}(i) = \mu s[a^s_i = a_i]$. Since $\mathbf{d}$ is a high c.e. degree, then by Martin's Theorem [5], there is a c.e. set $A \in \mathbf{d}$, with an enumeration $\{A_s\}_{s \in \omega}$ of A, such that the computation function of $\bar{A}$ is dominant. In other words, if g is a computable function, then $c_{\bar{A}}(i) > g(i)$ for almost every $i \in \omega$.

We will define a Turing functional Γ in stages such that at each stage s, we are computing X_s from the current approximation of A, A_s, with a certain associated use γ. To ensure that Γ is correct, once we've defined $\Gamma^{A_s}(z) = X_s(z)$ on some $z \in \omega$ with use $\gamma(z)$, we cannot cannot change the value of $X(z)$ until we have a change in our current oracle A_s below the use of the computation. Using this method, each time we see a realization and wish to enumerate a number into X, we must ask for permission

from A. In other words, in order to enumerate x into X, we need to ask A to change below the use $\gamma(x)$. The set A "gives us permission" to enumerate x if A enumerates a number $\leq \gamma(x)$ at some later stage.

Suppose we want to enumerate $x^i - 1$ to shift y_0 as desired. Then we must wait for A to enumerate a number below the use $\gamma(x^i - 1)$ to do so. Since the success of our construction depends on being able to enumerate when desired, we must ensure that A will give us permission to enumerate often enough. To secure our win, we will exploit the above property of A and amend our procedure so that receiving permission co-finitely often will be sufficient.

We will use this property by building a computable function g for our requirement Q_e that will challenge the domination of $c_{\bar{A}}$ and ensure that we will receive permission sufficiently often to succeed. The general idea is as follows: We define Γ by stages to compute X from the current approximation of A. We link each interval (x^{i-1}, x^i) to a member, say a_j, of the current complement of A by defining the use of the computation on this interval to be a_j. If we wish to change X in this interval, then we must wait for a change in A below a_j. We define our function g in order to tempt the desired change in A. Suppose at stage s, x^i is realized for Q_e. Then at this stage, we wish to enumerate the predecessor of x^i in the interval (x^{i-1}, x^i). We will define $g(j) = s$ and by doing so we are stating that we want a change in A below the use a_j after stage s. This definition is made in order to push A to change. Defining g in this manner ensures that since $c_{\bar{A}}(i) > g(i)$ for almost every $i \in \omega$, A can deny permission for only finitely many $j \in \omega$. Of course, permission does not have to be given immediately so we need to leave sufficient room in our construction to handle this.

We need to examine how this new development affects our strategy. If x^0 is realised at stage s, then we will ask permission to enumerate $x^0 - 1$ in order to shift our y_0. The interval $[0, x^0]$ will be linked to a certain member of $\bar{A}$, say a_j, and will thus be waiting on a change below a_j. It is possible that such a change will never occur after stage s. In this case, we will never receive permission to enumerate the predecessor of x^0 and will thus not shift y_0 as desired. Thus, if we rely solely on this y_0-strategy, our construction may not progress. To take this into account, once we ask permission from A on this interval, we will set it aside and continue our construction with the idea that

we may never get permission here. We will turn our attention to the larger interval $[0, x^1]$ and wait for a longer agreement.

Suppose at stage s', x^1 is realized. Then at this point we shift our attention to the apparent image of x^1, call this y_1. We now start a new strategy, hoping that y_1 will be the number that is shifted to the right of each x^i. We ask permission to enumerate $x^1 - 1$, and again wait for an appropriate change in A. We continue in this manner until we receive from A permission to enumerate into an interval, say (x^{i-1}, x^i). Then we will enumerate the predecessor of x^i thus forcing a shift of its apparent image, y_i, as desired. We will now pursue the stragegy for y_i by asking permission to enumerate and move y_i to the right of x^{i+1}, and abandon all other initiated y_j's to the right of our y_i. Since we want one winner in the end, we will maintain all y-strategies that are to the left of y_i as the intervals they are linked to are still intact. To ensure success of our construction, we will actually consider infinitely many possible y_i's and argue that, in the end, one of them must have infinitely many predecessors.

Since we are selecting infinitely many y_i's to move, we must ensure that we have enough room between each of the x^i so that we can enumerate on behalf of Q_e whenever we need to. We need to be able to enumerate a unique predecessor of x^i to shift each image y_j, $j \leq i$, past x^i. For example, we may realize x^0 and x^1 before receiving permission to enumerate below x^0. Thus we are waiting to enumerate both $x^0 - 1$ and $x^1 - 1$ into X. Suppose we first receive permission to enumerate $x^1 - 1$ to move y_1 to the right of x^1 and do so. Then we may receive permission to enumerate below $x^0 - 1$ at a later stage. We will then enumerate $x^0 - 1$ to move the apparent image y_0 of x^0 to the right. Indeed y_0 is now sitting under $x^1 - 1$ or x^1 in our construction and is now waiting to move past x^1. To do so, we must ask permission again to enumerate below x^1. If there is no more room in this interval, then y_0 is now stuck. To continue, we must be able to enumerate $x^1 - 2$ into X. Since y_1 has already moved to the right of x^1, once y_0 has shifted past x^1 we will never again need to enumerate in the interval (x^0, x^1). We thus require at most one number below x^0 and two numbers between x^0 and x^1. Following similar reasoning, we need at most $i + 1$ natural numbers between x^{i-1} and x^i, so choosing a function f with $i+1$ numbers between $f(i-1)$ and $f(i)$ will suffice.

Each time we act, we are enumerating the right most number in (x^{i-1}, x^i) that is not yet in X, so the numbers in $(\omega, \leq)$ that are in our set X will be found in blocks. In other words, for any $z \in X$, either $z = x^i$ for some i or the successor of z is also in X. There are countably many blocks, and each block has an x^i as its right-most member. We will call the block of elements in the interval $(x^{i-1}, x^i]$ that are in X the x^i-block. The important point is that if some y_j has been shifted to the right by our opponent, creating new length of agreement, then y_j must appear in the image of some x^i-block, $i > j$. Now let us suppose that x^i is realized at stage s. Then we wish to enumerate the number, say z, immediately to the left of the x^i block. (Note that such a number greater than x^{i-1} must exist since we have already supposed that we've left enough room to enumerate when we so desire.) Thus $z \notin X_s$ and $z_{L_e} \in V_e$. If at some stage we receive permission to enumerate z then z_{L_e} is now sitting under an element of X and cannot be enumerated into W_e. Since $[z, x^i] \subset X$, if the opponent fixes the problem then z_{L_e} must be to the right of the x^i-block thus forcing the image y_i of x^i to shift to the right of x^i. In fact, y_i must end up under one of the x_j-blocks to the right.

We will now outline a construction to satisfy a single requirement Q_e, and $X \leq_T A$.

One Requirement:

Construction 1:

Let $\mathbf{d}$ be a high c.e. degree. Let $A \in \mathbf{d}$, with an enumeration $\{A_s\}_{s\in\omega}$, such that $c_{\bar{A}}$ is dominant. Let $\bar{A}_s = \{a_0^s < a_1^s < a_2^s < \ldots\}$. We build a set X satisfying Q_e and define a Turing functional Γ by stages such that $\Gamma^A = X$. Let $x^0 = 1$ and for each $i \in \omega$, let $x^i = x^{i-1} + (i+1)$ and enumerate the set $\{x^i : i \in \omega\} \subset \omega$ into X. We will let γ be the use function for Γ and we will define a secondary function λ that we will call the link function. At stage 0, set $\lambda_0(i) = i$ for all $i \in \omega$ and let $\gamma_0(i) = a^0_{\lambda_0(i)} = a^0_i$ for each $i \in \omega$. In doing so, we are making our first definitions of Γ as follows: for each $z \in (x^{i-1}, x^i)$, we are computing $\Gamma^A(z) = X(z)[0]$ with use $\gamma(z) = a_{\lambda(i)}[0]$. In general, at any stage s, we are computing $\Gamma^A(z) = X(z)[s]$ for $z \in (x^{i-1}, x^i)$ with use $\gamma(z) = a_{\lambda(i)}[s]$. Now if at any point we have a change in A, for example, if A enumerates a_i^s, then any computation of Γ with use greater than or equal to a_i^s is lost. Let m be such that $\lambda_s(m) = i$, then

by the definition of the link function λ, $\Gamma^{A_s}(z)$ for $z \in (x^{m-1}, x^m)$ is computed with use a_i^s. Since A_s has had a change within the use of the computation, the computation is lost and we can now add a new definition, $\Gamma^{A_{s+1}}(z)$, with our new approximation of A. Hence we may enumerate any $z \in (x^{m-1}, x^m)$ into X.

We will define a computable function g on behalf of Q_e which will challenge the domination of A. Now suppose at stage s we have defined $g(0)$, $g(1)$,...,$g(k)$ of our computable function g. Assume that at the end of stage s, we were waiting for x^{i_0} to be realized. Consider the following 2 cases at stage $s+1$:

1. A enumerates a_i^s, and we have asked permission to enumerate below some x^m with $\lambda(m) \geq i$.

2. x^{i_0} is realized.

If we are in case 1, then we have received permission to change our computation of X on the intervals (x^{m-1}, x^m) for all m with $\lambda(m) \geq i$. Select the least such m for which we have defined $g(\lambda(m)[s])$. (I.e. the left-most interval on which we are awaiting permission from A.) We enumerate the number preceeding the current x^m-block into X. At this time, we abandon all j-strategies that are waiting on intervals to the right of x^m, while making sure to maintain all j-strategies whose intervals are still intact. We will now wait for x^{m+1} to be realized. Define $\lambda(m+1)[s+1] = k+1$, the first natural number where g has not yet been defined, and $\lambda(m+1+j)[s+1] = k+1+j$ for each $j \geq 1$. We have shifted our link function in order to ensure that $g(\lambda(m+1))$ is not defined at stage $s+1$. Define $\Gamma^A(z) = X(z)[s+1]$ for all $z \in (x^{i-1}, x^i)$ with use $\gamma(z) = a_{\lambda(i)}[s+1]$.

Now suppose we are in case 2. We now define $g(\lambda(i_0)) = s+1$ to ask permission to enumerate in the interval (x^{i_0-1}, x^{i_0}). Note that $g(\lambda(i_0))$ was undefined at the end of stage s. We are thus challenging A_s to enumerate below $a_{\lambda(i)}^{s+1} = a_{\lambda(i)}^s$. If at the last eventful stage t we enumerated into the interval (x^{i_0-2}, x^{i_0-1}) on behalf of some y_{j_0}, then this realization means that y_{j_0} must now be at least as far as the x^{i_0}-block at stage $s+1$. If instead we saw a realization at this stage t, then we must have seen x^{i_0-1} realized at that time. We would have then moved on to the interval $[0, x^{i_0}]$ with the

assumption that our lower j-strategies will not succeed. When we see x^{i_0} realized at stage $s+1$ we will want to initialize our y_{i_0}-strategy (possibly not for the first time). It is now clear that our earlier defintion of $g(\lambda(i_0)) = s+1$ was made in an attempt to eventually shift this y_{i_0} to the next x^{i_0+1}-block. In the meantime, we now turn our attention to x^{i_0+1} and await a realization or an advantageous enumeration from A. Note that at stage $s+1$, g is undefined on $\lambda(i_0+1)$.

If neither case occurs, then we have not received permission to enumerate anything into X at stage $s+1$, nor have we seen a new realization. We thus let $X_{s+1} = X_s$ and leave the link function λ unchanged. We may however have a new approximation of A, A_{s+1}, and so we define $\Gamma^A(z) = X(z)[s+1]$ for all $z \in (x^{i-1}, x^i)$ with use $\gamma(z) = a_{\lambda(i)}[s+1]$. Note that if $A_{s+1} = A_s$, then nothing changes.

End Construction 1

A proof of construction 1 is not provided here as we will prove a more detailed construction involving all requirements Q_e shortly. We can however provide some discussion.

By how we chose our x^i's, we know that whenever we wish to enumerate on behalf of Q_e, we have room to do so. We also defined our Turing functional Γ in stages so that in the limit $\Gamma^A = X$. To ensure that this happens, we must prove that the use of our computation settles and that one of our computations will hold forever. Since $\lambda(n)$ can only change finitely often, we can show that $a_{\lambda(n)}$ will eventually settle.

We must also show that Q_e will be satisfied. If $\langle L_e, W_e, V_e \rangle$ is only a threat finitely often, then Q_e is clearly satisfied. So it suffices to show that if there are infinitely many stages s where $L_{e,s}$ appears to be isomorphic to $(\omega, \leq, X_s)$ as computed by $(W_{e,s}, V_{e,s})$, then in the limit, L_e will have an element y with infinitely many predecessors. (This element y will be equal to y_j for some $j \in \omega$.) If infinitely many such stages exist, then by our construction, the computable function g that we build must be total. If g is total, then it must be dominated by the computation function of $\bar{A}$. This means that there must exist a stage after which A gives us permission whenever we ask. One of the y_j's that is "alive" at that time will be our winner: the element that is pushed past x^i for each $i \in \omega$. Since $y_j \in L_e$ has infinitely many predecessors, L_e cannot be

isomorphic to $(\omega, \leq)$ and thus Q_e is satisfied.

This construction is not the most efficient, and there are many precautions taken here that need not be in order to satisfy only one requirement. This was done so that only slight amendments must be done if we wish to include all requirements Q_e, $e \in \omega$. Also, in the previous construction we did not ensure that $A \leq_T X$ by using our master list $\{x^i : i \in \omega\} \subset \omega$. In this next construction, we will include this coding and and examine how the countable list of requirements interact.

All Requirements:

Before attempting construction 2 involving all requirements, we must examine the method of satisfying a single requirement, and study how all the requirements will interact. If there is injury, then we may need to amend Construction 1 to take that into account. Since no requirement is trying to keep numbers out of X on its behalf, there is no real interaction between the requirements. Each requirement Q_e is only concerned with 3 things: a length of agreement on a certain interval $[0, x^i]$, asking permission to enumerate below the x^i-block and enumerating as desired. The former two actions do not result in a change in X and thus do not affect the strategy of the other requirements. The latter case does result in a change in X, but we argue that this will not affect the other strategies either. Q_e is only allowed to enumerate the predecessor of an x^i-block into X, thus we are at worst lengthening that block by one number. We are not introducing any extraneous '1's strictly between our x^i blocks, so our enumeration strategy will remain unchanged. The only issue is ensuring that we now have enough room for each of the countably many requirements to enumerate whenever necessary. This will of course mean having more room between each of the x^i's in the new construction. Now we will describe our second construction in detail.

Instead of determining whether a single tuple $\langle L_e, W_e, V_e \rangle$ is a threat, we will now consider all 3-tuples simultaneously. This will allow us to consider all possible routes our opponent could take to win. To ensure that each requirement will have room to enumerate when they need to, we put some restrictions on all "lower priority" arguments. A requirement Q_i has higher priority than Q_j if and only if $i < j$. When running

the procedure, requirement Q_e will first look for an agreement on the interval $[0, x^e]$ and and each larger interval in turn. In other words, we will not consider Q_e a threat until we see an agreement between $\langle L_e, W_e, V_e \rangle$ and $(\omega, \leq, X)$ all the way up to x^e. For example, only Q_0 will wait for an agreement on $[0, x^0]$, only Q_0 and Q_1 will look for an agreement on $[0, x^1]$, etc. We will run our enumerations of $\langle L_e, W_e, V_e \rangle$, and we will construct our procedure so that at any given time we are only concerned with a finite approximation of each triple and a finite collection of these triples. In other words at a stage s, we consider finite approximations $\langle L_0, W_0, V_0 \rangle [s]$,...,$\langle L_s, W_s, V_s \rangle [s]$. We then check for an agreement according to our priority list and act in a similar manner as in Construction 1 if we find one. While the requirements are interacting, there is no injury since each requirement is waiting for the same event to occur: namely, for the predecessor of some x^i-block to be enumerated. There could be any finite number of requirements waiting for permission under a single x^i, and if we enumerate $x^i - 1$ on behalf of one of them, they all win.

To ensure that $A \leq_T X$, we will produce a second master list of numbers for coding. Along with our set $\{x^i : i \in \omega\} \subset \omega$ determined by an appropriate increasing computable function f, we will construct as second set $\{z^i = x^i + 1 : i \in \omega\} \subset \omega$ for coding. In our construction, we will enumerate z^i into X if and only if i goes into the set A. With this new development, we will enumerate each x^i into X as before, and we will have the usual x^i-blocks now succeeded by a single number z^i that may or may not be enumerated into X during our construction. If at any point in our construction z^i is enumerated into X to code A, z^i will officially become the leading member of the x^i-block. In a way, we are informally re-setting $x^i = x^i + 1 = z^i$ at this moment. Of course, the actual value of x^i, as given by the function f, does not change.

Since we now have countably many requirements, we need more room between each of the x^i's. In fact, we need to choose our computable function f such that f has $1+2+...+(i+1) = \frac{(i+1)(i+2)}{2}$ numbers between $x^{i-1}+1 = z^{i-1}$ and x^i. By construction 1, our highest requirement Q_0 needs one space to enumerate under x^0, two spaces between x^0 and x^1, etc. Based on our restrictions above, Q_1 does not enumerate below x^0, but requires one space to enumerate under x^1, two spaces between x^1 and x^2, etc. So we will require 1 space below x^0, $1 + 2 = 3$ spaces between z^0 and x^1, $1 + 2 + 3 = 6$

spaces between z^1 and x^2 and so on. Of course this computation only works if we abandon j-strategies of each requirement exactly as we did in construction 1. In our next construction, we will ensure that a j-strategy for requirement Q_e is only abandoned if we enumerate in a lower interval on behalf of Q_e. A sufficiently low enumeration for the sake of a different requirement will not result in abandoning any j-strategies of Q_e. With this independence in mind, each requirement Q_e, $e \in \omega$, will require one spot below x^e, two below x^{e+1}, etc. By this reasoning, the function described above will provide sufficient room for each requirement.

Each requirement Q_e will choose inifitely many potential $y(e)$'s just as the single requirement did in Construction 1. The goal for the following construction is to build a set $X \subset \omega$ such that for each requirement Q_e, either $\langle L_e, W_e, V_e \rangle [s]$ only appears to be isomorphic to $(\omega, \leq, X_s)$ for finitely many stages s, or L_e has a member, say $y(e)$, with infinitely many predecessors.

In this construction, all requirements will use the same link function λ and use function γ for our functional Γ; however, each requirement will have its own auxillary function g_e to challenge A to enumerate on its behalf. Because we now have so many requirements to keep track of, we will also build a tracking function $\tau_e : \omega \to \omega$ and a position function $\pi_e : \omega \times \omega \to \omega$ for each requirement. The function τ_e keeps track of what interval Q_e is waiting on at each stage. For example, if $\tau_e(s) = i$ then at stage s, we are waiting for x^i to be realized for Q_e. The function π_e follows the position of each j-strategy of Q_e. We note here that the j-strategy is initialized for the first time when we first see x^j realized for Q_e. We define $y_j(e)$ to be the image of x^j at this stage.

At the beginning of the construction, we will set $\pi_e(j, 0) = -1$ for each $j \in \omega$. When we initialize the j-strategy for Q_e we will set $\pi_e(j) \geq 0$. If $\pi_e(j, s) = -1$ then at stage s the j-strategy for Q_e is not active. This will be the case either if x^j has not yet been realized (for the first time) by stage s, or if we have recently enumerated (for Q_e) into a low enough interval to abandon the j-strategy. If at stage s we set $\pi_e(j, s) = i \geq 0$, then at this stage either $y_j(e)$ is waiting on an enumeration in the interval (x^{i-1}, x^i), or we have just enumerated in this interval on behalf of $y_j(e)$ and are waiting for x^{i+1} to be realized for Q_e. By our construction, only finitely many $j(e)$'s will have non-negative π_e values at any given time for each requirement Q_e.

Construction 2:

Let $\mathbf{d}$ be a high c.e. degree. Let $A \in \mathbf{d}$, with an enumeration $\{A_s\}_{s\in\omega}$, such that $c_{\bar{A}}$ is dominant. Let $\bar{A}_s = \{a_0^s < a_1^s < a_2^s < \ldots\}$. We build a set X satisfying Q_e for each e and define a Turing functional Γ by stages such that $\Gamma^A = X$. Let $x^0 = 1$, $z^0 = x^0 + 1 = 2$ and for each $i \in \omega$, let $x^i = z^{i-1} + \frac{(i+1)(i+2)}{2}$ and $z^i = x^i + 1$. Enumerate the set $\{x^i : i \in \omega\} \subset \omega$ into X.

We will let γ be the use function for Γ and let λ be the link function. At stage 0, set $\lambda_0(i) = i$ for all $i \in \omega$ and let $\gamma_0(i) = a^0_{\lambda_0(i)} = a^0_i$ for each $i \in \omega$. In general, at any stage s, we are computing $\Gamma^A(z) = X(z)[s]$ for $z \in (x^{i-1}, x^i)$ with use $\gamma(z) = a_{\lambda(i)}[s]$. Every requirement will be using the same link function and the same use of the computation. We will build a separate comuptable function g_e for each requirement Q_e, each challenging the domination of $c_{\bar{A}}$.

Finding lengths of agreement is a similar process as in Construction 1, but we can now have x^i realized for any Q_e with $i \geq e$. At each stage, we are looking for the smallest i such that x^i is realized for some Q_e (could be more than one), and this defines the interval where we will ask for a new permission from A.

At stage 0, define $\tau_0(0) = 0$ and $\pi_e(j, 0) = -1$ for all $e, j \in \omega$. Suppose we are at stage $s + 1$. To begin, define $\tau_{s+1}(s + 1) = s + 1$. At stage $s + 1$, two distinct events can occur:

1. A enumerates a_i^s, and at least one requirement has asked permission to enumerate below some x^m with $\lambda(m) \geq i$.

2. x^i is realized for some Q_e ($e \leq s$) with $\tau_e(s) = i$.

Suppose we are in case 1. First enumerate $z^{a_i^s}$ into X for coding. Note that $z^{a_i^s} = x^{a_i^s} + 1$ is now the leader of the $x^{a_i^s}$-block. For each $e \leq s$ determine whether $g_e(\lambda(m))$ has been defined for some m with $\lambda(m) \geq i$. If so, pick the least m_e with this property for each e. This list refers to the left-most intervals on which each requirement Q_e is awaiting permission. Enumerate the number preceeding the x^{m_e}-block into X for each

such m_e. Note that we only enumerate one number below each x^{m_e}, even if $m_{e_1} = m_{e_2}$ for two different requirements Q_{e_1} and Q_{e_2}. By our choice of x^{m_e} and x^{m_e-1} (determined by our function f), we know that there is enough room to enumerate below each x^{m_e}-block at stage $s+1$. Let $\tau_e(s+1) = m_e + 1$ for each such e to state that Q_e is now waiting for x^{m_e+1} to be realized.

Let m be the least m_e as described above. Let

$$N = \max\{n \in \omega : g_e(n) \text{ is defined for some } e\}.$$

Define $\lambda(m+1)[s+1] = N+1$ and $\lambda(m+j)[s+1] = N+1+j$ for each $j \geq 1$. Doing so ensures that g_e is not defined on $\lambda(\tau_e(s+1)) = \lambda(m_e+1)$ for any e (from above) at stage $s+1$. Define $\Gamma^A(z) = X(z)[s+1]$ for all $z \in (x^{i-1}, x^i)$ with use $\gamma(z) = a_{\lambda(i)}[s+1]$. Lastly, we need to define $\pi_e(j, s+1)$ for all j. If Q_e did not receive permission to enumerate, then let $\pi_e(j, s+1) = \pi_e(j, s)$ for all j. If we enumerated below x^{m_e} on behalf of Q_e then there must exist a unique number j_e such that $\pi_e(j_e, s) = m_e$. For all $j \leq j_e$, let $\pi_e(j, s+1) = \pi_e(j, s)$, and for all $j > j_e$, let $\pi_e(j, s+1) = -1$.

Suppose case 2 occurs. Let i_0 be the smallest i such that x^i is realized for some Q_e ($e \leq s$) with $\tau_e(s) = i$. Find all requirements Q_e, $e \leq s$, such that $\tau_e(s) = i_0$ and x^{i_0} is realized for Q_e at stage $s+1$. There are finitely many such requirements, so let them be $Q_{e_1}, \ldots, Q_{e_k}$. For each $i \in \{1, \ldots, k\}$, define $g_{e_i}(n) = s+1$ for all $n \leq \lambda_s(i_0)$ for which $g_{e_i}(n)$ has not yet been defined. This is done first to ensure that g_e will indeed be total if there are infinitely many stages where Q_e is a threat, and second to ask permission to enumerate in the interval (x^{i_0-1}, x^{i_0}) on behalf of each of the Q_{e_i}'s. Note that g_{e_i} has not been defined on $\lambda(\tau_{e_i}(s))$ by the end of stage s, for each $i \in \{1, \ldots, k\}$.

For each e_i ($i = 1, \ldots, k$) do the following: Determine whether or not there exists some $j_0 < i_0$ with $\pi_{e_i}(j_0, s) = i_0 - 1$. If so, let t be the last stage when we either enumerated into X on behalf of Q_{e_i} or saw some x^l realized for Q_{e_i}. If $\tau_{e_i}(t-1) \geq \tau_{e_i}(t)$, then at stage t we enumerated some number into X. (Note that $\tau_{e_i}(t) = \tau_{e_i}(s)$.) In this case we let $\pi_{e_i}(j_0, s+1) = \pi_{e_i}(j_0, s) + 1 = i_0$ to shift our j_0-strategy for Q_{e_i} to the interval (x^{i_0-1}, x^{i_0}). This definition reflects the fact that this y_{j_0} is now at least as far as the x^{i_0}-block since we have recovered up to x^{i_0} after having enumerated below x^{i_0-1}.

If $\tau_{e_i}(t-1) < \tau_{e_i}(t)$, then we necessarily saw x^{i_0-1} realized for Q_{e_i} at stage t. In this case, we let $\pi_{e_i}(i_0, s+1) = i_0$ to initialize y_{i_0} and pursue the new i_0-strategy for Q_{e_i}.

Leave all other definitions of π_e and τ_e unchanged at stage $s+1$.

If neither case occurs, then we have not received permission to enumerate anything into X at stage $s+1$, nor have we seen a new realization for any requirement. We thus let $X_{s+1} = X_s$ and leave the link function λ unchanged. We may however have a new approximation of A, A_{s+1}, and so we define $\Gamma^A(z) = X(z)[s+1]$ for all $z \in (x^{i-1}, x^i)$ with use $\gamma(z) = a_{\lambda(i)}[s+1]$. Note that if $A_{s+1} = A_s$, then nothing changes. Finally, let $\tau_e(s+1) = \tau_e(s)$ and $\pi_e(j, s+1) = \pi_e(j, s)$ for all $e, j \in \omega$.

End Construction 2

Verification:

We defined our Turing functional Γ in stages so that in the limit $\Gamma^A = X$; first we must prove that the use of our computation settles.

Lemma 2.4.1. $\lim_{s\to\infty} \gamma_s(n) < \infty$ *for each* $n \in \omega$.

Proof. To prove this, it is enough to show that the link function λ settles on each number. The link function λ only changes when we enumerate into X. Specifically, $\lambda(n)$ only changes when we enumerate into the interval $(0, x^n)$. Since there are finitely many numbers below x^n, $\lambda(n)$ can only change a finite number of times. Since A is noncomputable, A is infinite and coinfinite, and hence if $\lambda(n)$ settles then so does $a_{\lambda(n)}$. Thus $\lim_{s\to\infty} \gamma_s(n) = \lim_{s\to\infty} a^s_{\lambda(n)} < \infty$. □

We get $A \leq_T X$ since $x^i + 1 \in X$ if and only if $i \in A$. By Lemma 2.4.1, the use settles and thus one of our computations hold forever with $\Gamma^A = X$. Hence $X \equiv_T A$ as desired.

It remains to show that Q_e is satisfied for all $e \in \omega$. To do this we need to show that for each e, either Q_e has only finitely many stages s where $\langle L_e, W_e, V_e\rangle[s]$ is a threat, or that in the limit, L_e has an element $y(e)$ with infinitely many predecessors. Recall

that we define $y_j(e)$ to be the apparent image of x^j the first time we see x^j realized for Q_e.

Lemma 2.4.2. *Fix $e \in \omega$. If $\pi_e(j,s) = i > 0$ then $y_j(e)$ is at least as far right as the image of the x^i-block in L_e at stage s.*

Proof. Fix a number $j \in \omega$. We will proceed by induction on the stage number s. Let s^e_j be the first stage where we see x^j realized for Q_e. Note that $\pi_e(j,s) = -1$ for all $s < s^e_j$. By our construction, at stage s^e_j we define $\pi_e(j, s^e_j) = j$ to initialize $y_j(e)$. Since $y_j(e)$ is defined to be $x^j_{L_{e,s^e_j}}$, we indeed have $y_j(e)$ in the image of the x^j-block at stage s^e_j, and hence the base case of our Lemma holds. Before we move on to our inductive step, we make the following obervation:

$$(*) \quad \forall z \in \omega \ (t_1 < t_2 \Rightarrow z_{L_{e,t_2}} \leq_{L_e} z_{L_{e,t_1}}).$$

This follows from the fact that L_e is c.e. so the number of predecessors that any member of the ordering has can only increase.

Now suppose that at stage $s > s^e_j$, we have $\pi_e(j,s) = i > 0$. Suppose that $\pi_e(j, s-1) = \pi_e(j,s) = i$. Then by the inductive hypothesis, $y_j(e)$ is at least as far as the x^i-block by stage $s-1$. Since $s > s^e_j$, by $(*)$ we have that $y_j(e)$ is at least as far as the x^i-block by stage s. Now suppose that $\pi_e(j, s-1) \neq \pi_e(j,s)$. Since the value of π_e can only change after Q_e enumerates or we see a realization for Q_e, we have two possibilities. Either $\pi_e(j, s-1) = -1$ and $\pi_e(j,s) = j$, or $\pi_e(j, s-1) = i-1$ and $\pi_e(j,s) = i$. (Of course we could also set $\pi_e(j)$ back to -1, but this doesn't not satisfy the statement of the Lemma.)

If we have the former possibility, then at stage s, we saw x^j realized for Q_e. (Note that here $i = j$) Since $s > s^e_j$, then by assumption, this is not the first time we are seeing this realization. At this moment we wish to claim that $y_j(e) \geq_{L_e} x^j_{L_e}$ at stage s. The number $y_j(e)$ was defined as $x^j_{L_{e,s^e_j}}$ at stage $s^e_j < s$ so by (*), $x^j_{L_{e,s}} \leq_{L_e} x^j_{L_{e,s^e_j}} = y_j(e)$ as claimed. Thus $y_j(e)$ is at least as far as the x^j-block of L_e at stage s.

In the latter case, at stage s we saw x^i realized for Q_e, and at the last stage $t < s$ when Q_e acted, we enumerated the predecessor of the x^{i-1}-block into X to pursue a

strategy for Q_e. Note that by assumption $\pi_e(j,t) = \pi_e(j,s-1) = i-1$. By the inductive hypothesis we must have $y_j(e)$ at least as far as the x^{i-1}-block at stage t. If $y_j(e)$ is already as far as the x^i-block at stage t, then we win, so let us assume that $y_j(e)$ is strictly to the left of the x^i-block of L_e at stage t.

Suppose that (A): $y_j(e) \leq_{L_e} x^{i-1}_{L_e}$ at stage t. Then $y_j(e)$ is necessarily in the x^{i-1}-block of L_e at this stage. We know that at stage t we enumerated the predecessor of this block into X on behalf of Q_e, thus introducing a new element, say z, into X below the preimage of $y_j(e)$ at this stage. Later, at stage s, we see x^i realized. This means that $L_{e,s}$ must have a block of 1s matching up with the x^{i-1}-block and the x^i-block in X. Since $z \in X$ and $z_{L_{e,t}} \notin X_{L_e}$, the opponent will have had no choice but to match $z_{L_{e,t}}$ with a '0' to the right of z. Since the x^{i-1}-block is contained in X, the next possible match for $z_{L_{e,t}}$ is strictly greater than x^{i-1}. Indeed, the next possible '0' is z^{i-1} if $z^{i-1} \notin X$ at stage s, and strictly greater than z^{i-1} otherwise. Since $y_j >_{L_e} z_{L_{e,t}}$ and $y_j \in X_{L_e}$ the next possible match for $y_j(e)$ lies in the x^i-block. We can thus conclude that at stage s, $y_j(e)$ must be at least as far as the x^i-block of L_e.

Now suppose that (B): $y_j(e) > z^{i-1}_{L_e}$ (but below the x^i-block) at stage t. Then $y_j(e)$ is necessarily matched with a '0' at this stage. When x^i is realized for Q_e at stage s, $y_j(e)$ must be matched with a '1' since $y_j(e) \in X_{L_e}$. The closest member of X is indeed in the x^i-block, so $y_j(e)$ has no choice but to be at least as far as the x^i-block at stage s.

Now let's assume that $y_j(e) = z^{i-1}_{L_{e,t}}$. If z^{i-1} has been enumerated into X by stage t, then the same reasoning from (A) will work here. If z^{i-1} is not in X by stage s, then the argument from (B) works. So without loss of generality, assume that z^{i-1} is in X_s but not X_t. At stage t we enumerated the predecessor of the x^{i-1}-block, say w, to destroy an agreement. This enumeration will induce a mismatch since $w_{L_{e,t}} \notin X_{L_{e,t}}$. In order for x^i to be realized for Q_e, this '0' has no choice but to move past the block of '1's to its right, and must move at least as far as z^{i-1} (since this is the next possible '0' in L_e) regardless of whether or not z^{i-1} is in X at this point. Since $y_j(e) = z^{i-1}_{L_{e,t}}$, we must have $y_j(e) \geq z^{i-1}_{L_{e,s}}$ in order for x^i to be realized. Moreover, since the next memeber of X is in the x^i-block and $y_j(e) \in X_{L_e}$, $y_j(e)$ must be at least as far as the x^i-block of L_e by stage s as desired. This completes the proof of Lemma 2.4.2. $\square$

Finally, we will show that if, in the end, our function g_e is total for some e, then one of the $y_j(e)$'s must be to the right of each x^i.

Lemma 2.4.3. *If g_e ($e \in \omega$) is total, then there exists some $j \in \omega$ with*

$$\lim_{s\to\infty} \pi_e(j, s) = \infty.$$

Proof. Fix $e \in \omega$. Let us assume that g_e is total. Since $c_{\bar{A}}$ is dominant and g_e is computable, we have $c_{\bar{A}}(i) > g_e(i)$ for almost every $i \in \omega$. Hence there exists some number k_e such that $c_{\bar{A}}(i) > g_e(i)$ for all $i > k_e$. Let s_0 be the stage where we define $g_e(k_e)$ (i.e. $g(k_e) = s_0$). Then after this stage, any time we define g_e on a new natural number to challenge $c_{\bar{A}}$, A must eventually give Q_e permission to enumerate.

Suppose that $\tau_e(s_0) = i$. Let s_1^e be the last stage when we enumerate below x^{i-1} for the sake of Q_e. Since there are only finitely many numbers in the interval $(0, x^{i-1})$, such a stage must exist. Now we may receive permission from A to enumerate below x^{i-1} for other requirements, but by our construction, these events can only happen finitely often and moreover will not change the values of $\pi_e(j)$ for any j. Let t_0^e be the next stage when x^i is realized for Q_e. This stage must exist since g_e is total. At stage t_0^e we will define $\pi_e(j_0, t_0^e) = i$ for some $j_0 \leq i$ to pursue the $y_{j_0}(e)$-strategy and define $g_e(\lambda(i)) = t_0^e$ to ask permission to enumerate below x^i on behalf of $y_{j_0}(e)$. At this point we set $\tau(t_0^e) = i + 1$. We now claim that next time $\pi_e(j_0)$ changes, say at stage t_1^e, we will have $\pi_e(j_0, t_1^e) = i + 1$.

By our construction, $\pi_e(j_0)$ will not change until after we receive permission to enumerate somewhere below x^i on behalf of Q_e. Also by our construction, at this later stage when $\pi_e(j_0)$ changes, it will either be set back to -1 or to $i + 1$, depending on the interval of enumeration. We would like to rule out the former case and ensure that the latter will indeed occur. Since $\lambda(i) > k_e$, A must eventually give us permission to enumerate below x^i for Q_e. By assumption, we no longer enumerate below x^{i-1} and thus this enumeration can only occur in the interval (x^{i-1}, x^i). Hence the value of $\pi_e(j_0)$ will not be set back to -1.

In Case 1 of our construction, we ensure that if A gives us permission to enumerate, then we indeed enumerate on behalf of each requirement Q_e that was waiting. In other words, if Q_e receives permission to enumerate into the interval (x^{i-1}, x^i) at some stage s', then we will enumerate in the desired interval at that stage on behalf of Q_e; this will happen regardless of whether or not another requirement received permission on a lower interval. We can thus conclude that this enumeration must happen for the sake of Q_e. At this point we will set τ_e to $i+1$ and wait for x^{i+1} to be realized for Q_e. Again, since g_e is total, we must see a later stage t_1^e where x^{i+1} is realized for our requirement. Following our construction, $\pi_e(j_0)$ will remain unchanged until this event occurs. At stage t_1^e we will define $\pi_e(j, t_1^e) = i + 1$ as desired, following Case 2 of Construction 2. Moreover, we know that from this point on, we are finished enumerating below x^i on behalf of requirement Q_e. The reasoning is as follows: By assumption, we will never again enumerate below x^{i-1} for the sake of Q_e, so no j-strategies of Q_e with $j < j_0$ will advance after stage t_0^e. Thus the only j-strategy that can ask for an enumeration in the interval $(0, x^i)$ for Q_e is the j_0-strategy. Since we have already enumerated below x^i on behalf of $y_{j_0}(e)$ for the last time, we will not enumerate below x^i for the sake of requirement Q_e after stage t_1^e.

Now suppose that we are at a stage $t_k^e > t_1^e$ with $\pi_e(j_0, t_k^e) = i + k$, and that we are finished enumerating below x^{i+k-1} on behalf of Q_e at stage t_k^e. We claim that there exists $t_{k+1}^e > t_k^e$ with $\pi_e(j_0, t_{k+1}^e) = i + k + 1$ and the property that we never again enumerate below x^{i+k} for Q_e after this stage. By similar reasoning as above, $\pi_e(j_0)$ will remain unchanged until we enumerate below x^{i+k} for the sake of Q_e. By assumption, we must see a later stage where A gives us permission to enumerate below x^{i+k} on behalf of Q_e, and this enumeration will indeed occur by Case 1 of our construction. Since this enumeration cannot take place to the left of x^{i+k-1}, it must be in the interval (x^{i+k-1}, x^{i+k}). By Case 1 of our construction, this enumeration will result in setting τ_e of this stage to $i + k + 1$, signifying that we are now waiting for x^{i+k+1} to be realized for Q_e. Since g_e is total, we must see a later stage $t_{k+1}^e > t_k^e$ where x^{i+k+1} is realized for Q_e. By Case 2 of our construction, we will set $\pi_e(j_0, t_{k+1}^e) = i + k + 1$ as desired. Since by assumption we are done enumerating below x^{i+k-1} for Q_e and we have already enumerated below x^{i+k} on behalf of the j_0-strategy of Q_e, we will never again ask to

enumerate in the interval $(0, x^{i+k})$ for Q_e after stage t^e_{k+1}. This proves our claim.

As a result, we can find stages $t^e_1 < t^e_2 < t^e_3 < ...$ such that $\pi_e(j_0, t^e_k) = i + k$ for each $k \in \omega$. So we have $\lim_{s\to\infty} \pi_e(j_0, s) \geq \lim_{k\to\infty} \pi_e(j_0, t^e_k) = \infty$ as desired. □

Q_e is satisfied for each $e \in \omega$:

Fix $e \in \omega$. If the function g_e is not total, then at some point we stopped seeing a new length of agreement for $\langle L_e, W_e, V_e\rangle[s]$, so Q_e is clearly satisfied. If g_e is total, then by Lemma 2.4.3, there exists a number $j \in \omega$ with $\lim_{s\to\infty} \pi_e(j, s) = \infty$. By how π_e is defined in our construction, we can conclude that there exists stages $s^e_1, s^e_2, s^e_3, ...$ such that $\pi_e(j, s^e_i) = i$ for all $i \in \omega$. By Lemma 2.4.2, we must have $y_j(e)$ at least as far as the x^i-block at stage s^e_i, so in the limit, we must have $y_j(e)$ to the right of each x^i. This shows that L_e contains an element with infinitely many predecessors, namely $y_j(e)$, and thus L_e cannot be isomorphic to $(\omega, \leq)$.

In either case Q_e is satisfied.

Chapter 3

Degree Spectra on $(\omega, \leq_\zeta)$

After having examined known results about the degree spectra of unary relations on $(\omega, \leq)$, we wish to extend these results to the structure $(\omega, \leq_\zeta)$. Passing to the integers, we lose the rigidity of the structure, and we can ask whether this will affect the degree spectrum of a relation. We can view $(\omega, \leq_\zeta)$ as a copy of $(\omega^*, \leq^*)$ (the natural numbers in reverse order) followed by a copy of $(\omega, \leq)$ and so it makes sense to ask whether there is a strong relationship between the degree spectrum of a given relation on $(\omega, \leq)$ and the degree spectrum of the corresponding relation on $(\omega, \leq_\zeta)$.

Most of our results from Chapter 2 relied on the fact that X was an infinite and coinfinite set; we needed infinitely many bit alternations on the infinite segment of $(\omega, \leq)$. It seems likely that the success of analogous theorems for the structure $(\omega, \leq_\zeta)$ will rely on a similar property of our set X: whether X is infinite and coinfinite on both segments, one segment or neither segment of $(\omega, \leq_\zeta)$.

If X has no infinite and coinfinite segment in $(\omega, \leq_\zeta)$, then the degree spectrum is trivial. Let us assume, without loss of generality, that X is finite on the left segement and cofinite on the right segment. Then there exist numbers a and b such that

$$x \leq_\zeta b \Rightarrow x \in X \quad \text{and} \quad x \leq_\zeta a \Rightarrow x \notin X$$

Then we are left with only a finite area of interest in $(\omega, \leq_\zeta)$, namely the segment between a and b. Because of this, $X_{\mathbf{L}}$ must be computable no matter what computable

copy $\mathbf{L}$ of $(\omega, \leq_\zeta)$ we choose. In other words, the degree spectrum of X is a singleton, namely $\{\mathbf{0}\}$.

It is left for us to distinguish between unary relations that have two infinite and coinfinite segments or simply one such segment in $(\omega, \leq_\zeta)$. For clarity, we will consider the usual copy of the integers with domain $\mathbb{Z}$ under the usual $\leq_\mathbb{Z}$-order:

$$\ldots \quad -3 \quad -2 \quad -1 \quad 0 \quad 1 \quad 2 \quad 3 \quad \ldots$$

This structure is computably isomorphic to our previous structure, $(\omega, \leq_\zeta)$, and hence the two structures can be identified for the purposes of determining the degree spectra of unary relations. For the time being, we will consider any unary relation X on $(\omega, \leq_\zeta)$ as a subset of $\mathbb{Z}$ (which, of course, can be identified with a subset of $\mathbb{N}$ under our coding.) This will allow us to differentiate between unary relations that form subsets of $\mathbb{Z}$ and those that form subsets of $\mathbb{N}$. The latter type can only be infinite and coinfinite on the right segment of $(\omega, \leq_\zeta)$.

Let $X \subseteq \mathbb{N}$. If $deg(X) = \mathbf{d}$ then we have $\mathbf{d} \in DgSp_{(\omega, \leq_\zeta)}(X)$ and $\mathbf{d} \in DgSp_{(\omega, \leq)}(X)$, since X has degree $\mathbf{d}$ on both standard copies:

$(\omega, \leq, X)$				0	$\boxed{1}$	2	$\boxed{3}$	4	5	$\boxed{6}$	...
$(\omega, \leq_\zeta, X)$	...	-3	-2	-1	0	$\boxed{1}$	2	$\boxed{3}$	4	5	$\boxed{6}$	...

We now claim that if X is any unary relation on $(\omega, \leq_\zeta)$ such that $X \subseteq \mathbb{N}$, then $DgSp_{(\omega, \leq_\zeta)}(X) = DgSp_{(\omega, \leq)}(X)$.

We will start by showing that for $X \subseteq \mathbb{N}$, $DgSp_{(\omega, \leq)}(X) \subseteq DgSp_{(\omega, \leq_\zeta)}(X)$.

Let $\mathbf{d} \in DgSp_{(\omega, \leq)}(X)$. Then, by definition, there is a computable presentation $\mathbf{L}$ of $(\omega, \leq)$ such that $deg(X_\mathbf{L}) = \mathbf{d}$. Using the program that builds $\mathbf{L}$ from $(\omega, \leq)$ we can build a computable copy, $\tilde{\mathbf{L}}$, of $(\omega, \leq_\zeta)$ such that the right segment of $\tilde{\mathbf{L}}$, $\tilde{\mathbf{L}}_{\geq 0}$, is computably isomorphic to $\mathbf{L}$. For example, let $\mathbf{L}$ and $X_\mathbf{L}$ be as follows:

$(\mathbf{L}, X_{\mathbf{L}})$					5	$\boxed{0}$	1	$\boxed{9}$	4	13	$\boxed{2}$	...
$(\tilde{\mathbf{L}}, X_{\tilde{\mathbf{L}}})$	...	-3	-2	-1	5	$\boxed{0}$	1	$\boxed{9}$	4	13	$\boxed{2}$	...

We thus have a clear effective procedure to obtain the ordering $\tilde{\mathbf{L}}$ from $\mathbf{L}$. Since $X_{\tilde{\mathbf{L}}}$ is contained on the right segment of $\tilde{\mathbf{L}}$ and this segment is computably isomorphic to $\mathbf{L}$, we must have $deg(X_{\tilde{\mathbf{L}}}) = deg(X_{\mathbf{L}}) = \mathbf{d}$. And hence $\mathbf{d} \in DgSp_{(\omega,\leq_\zeta)}(X)$.

We will now argue that the reverse inclusion also holds, that is, $DgSp_{(\omega,\leq_\zeta)}(X) \subseteq DgSp_{(\omega,\leq)}(X)$ for such a set X.

Let $X \subseteq \mathbb{N}$ and suppose that $\mathbf{d} \in DgSp_{(\omega,\leq_\zeta)}(X)$. Consider the computable copy, $\tilde{\mathbf{L}}$, of $(\omega, \leq_\zeta)$ having $deg(X_{\tilde{\mathbf{L}}}) = \mathbf{d}$. For example:

$(\omega, \leq_\zeta, X)$	...	-4	-3	-2	-1	0	$\boxed{1}$	2	$\boxed{3}$	4	5	$\boxed{6}$	...
$(\tilde{\mathbf{L}}, X_{\tilde{\mathbf{L}}})$	...	7	-1	-9	2	0	$\boxed{6}$	-2	$\boxed{1}$	11	3	$\boxed{\text{-13}}$	...

Note that we have kept $0_{\tilde{\mathbf{L}}} = 0$ for clarity. Then for any $n \in \omega$, $\tilde{\mathbf{L}}_{\geq n} \cong (\omega, \leq)$ and, in particular, $\tilde{\mathbf{L}}_{\geq 0} \cong (\omega, \leq)$. Now we wish to build a computable copy $\mathbf{L}$ of $(\omega, \leq)$ that is computably isomorphic to the right segment of $\tilde{\mathbf{L}}$, $\tilde{\mathbf{L}}_{\geq 0}$.

Since $X \subseteq \mathbb{N}$, we know that no member of $X_{\tilde{\mathbf{L}}}$ can appear to the left of 0, so we will build in the ordering $\mathbf{L}$ only if $\tilde{\mathbf{L}}$ builds to the right of 0. For example, when -1 appears in $\tilde{\mathbf{L}}$ (to the left of 0), we will not introduce a match for -1 in $\mathbf{L}$. However, when 1 appears in $\tilde{\mathbf{L}}$ (to the right of 0), we will place the next natural number into $\mathbf{L}$ as the isomorphic image $\varphi(1)$ in $\mathbf{L}$. We reveal $\tilde{\mathbf{L}}$ stage by stage and build the ordering $\mathbf{L}$ as follows:

Stage 0 :	$\tilde{\mathbf{L}}$			0			
	$\mathbf{L}$			0			
Stage 1 :	$\tilde{\mathbf{L}}$			0	1		
	$\mathbf{L}$			0	1		
Stage 2 :	$\tilde{\mathbf{L}}$		-1	0	1		
	$\mathbf{L}$			0	1		
Stage 3 :	$\tilde{\mathbf{L}}$	-1	2	0	1		
	$\mathbf{L}$			0	1		
Stage 4 :	$\tilde{\mathbf{L}}$	-1	2	0	-2	1	
	$\mathbf{L}$			0	2	1	

Since we do not change our minds about the definition of φ, the isomorphism between $\tilde{\mathbf{L}}_{\geq 0}$ and $\mathbf{L}$ is a computable function. Since the $\tilde{\mathbf{L}}_{\geq 0}$ and $\mathbf{L}$ are computably isomorphic, any relation on $\tilde{\mathbf{L}}_{\geq 0}$ and its isomorphic image in $\mathbf{L}$ must have the same Turing degree. Since $X_{\tilde{\mathbf{L}}}$ is contained in $\tilde{\mathbf{L}}_{\geq 0}$, we must have $deg(X_{\mathbf{L}}) = deg(X_{\tilde{\mathbf{L}}}) = \mathbf{d}$, and hence $\mathbf{d} \in DgSp_{(\omega,\leq_\zeta)}(X)$ as desired.

We now know that for any $X \subseteq \mathbb{N}$, we have $DgSp_{(\omega,\leq_\zeta)}(X) = DgSp_{(\omega,\leq)}(X)$. It remains to determine whether an abitrary $X \subseteq \mathbb{Z}$ will have the same property, or do we introduce new possible degree spectra that can be realized in this case? In other words, do we get anything new if we allow the set X to distribute as an infinite and coinfinite set on both segments of $(\omega, \leq_\zeta)$? While this question is open and an answer will not be presented in this paper, we will provide some insight on how to reduce the problem of determining degree spectra on $(\omega, \leq_\zeta)$ to a question about degree spectra on $(\omega, \leq)$.

Let $X \subseteq \mathbb{Z}$. Since we wish to discuss Turing degrees of sets, we will return to the structure $(\omega, \leq_\zeta)$ under our coding

$$\ldots \ 9 \ \ 7 \ \ 5 \ \ 3 \ \ 1 \ \ 0 \ \ 2 \ \ 4 \ \ 6 \ \ 8 \ \ 10 \ \ \ldots$$

so that the domain is again ω. Recall that now the positive numbers in $\mathbb{Z}$ are coded as even numbers and the negative numbers are coded as odd numbers. We will consider an example to illustrate the next result:

Let us suppose that X distributes over the standard copy of $(\omega, \leq_\zeta)$ as follows:

$$(\omega, \leq_\zeta, X) \qquad \ldots 9 \ \ \boxed{7} \ \ \boxed{5} \ \ 3 \ \ \boxed{1} \ \ 0 \ \ \boxed{2} \ \ \boxed{4} \ \ 6 \ \ \boxed{8} \ \ 10 \ldots$$

Recall that the join of two sets A and B, denoted $A \oplus B$ is defined as

$$A \oplus B = \{2n+1 : n \in A\} \cup \{2n : n \in B\}.$$

Let $X_1 = \{\frac{x-1}{2} : x \in X, x \ odd\} = \{0, 2, 3, ...\}$ and $X_2 = \{\frac{x}{2} : x \in X, x \ even\} = \{1, 2, 4, ...\}$. Then the degree of X_1 is equal to the degree of $X \cap \{2n+1 : n \in \mathbb{N}\}$ and the degree of X_2 is equal to the degree of $X \cap \{2n : n \in \mathbb{N}\}$, and we have $X = X_1 \oplus X_2$.

Since the right and left segments of $(\omega, \leq_\zeta)$ are isomorphic to the structures $(\omega, \leq)$ and $(\omega^*, \leq^*)$ respectively, we can consider $X_1, X_2 \subset \mathbb{N}$ as unary relations on $(\omega, \leq)$ and $(\omega^*, \leq^*)$ as follows:

$$(\omega, \leq, X_2) \qquad\qquad 0 \ \ \boxed{1} \ \ \boxed{2} \ \ 3 \ \ \boxed{4} \ \ 5 \ \ \ldots$$

$$(\omega^*, \leq^*, X_1) \qquad \ldots \ \ 5 \ \ 4 \ \ \boxed{3} \ \ \boxed{2} \ \ 1 \ \ \boxed{0}$$

And we get the following proposition:

Proposition 3.1. *Let X be unary relation on $(\omega, \leq_\zeta)$ and let $X = X_1 \oplus X_2$ as previously described. Then*

$$DgSp_{(\omega, \leq_\zeta)}(X) = \{\boldsymbol{d}_1 \cup \boldsymbol{d}_2 : \boldsymbol{d}_1 \in DgSp_{(\omega, \leq)}(X_1) \text{ and } \boldsymbol{d}_2 \in DgSp_{(\omega, \leq)}(X_2)\}$$

where $d_1 \cup d_2$ denotes the join of the degrees d_1 and d_2.

We will denote the right hand side as $DgSp_{(\omega, \leq)}(X_1) \oplus DgSp_{(\omega, \leq)}(X_2)$.

Proof. Let $X = X_1 \oplus X_2$ as defined above. First we will show that

$$DgSp_{(\omega,\leq)}(X_1) \oplus DgSp_{(\omega,\leq)}(X_2) \subseteq DgSp_{(\omega,\leq_\zeta)}(X).$$

Suppose that $\mathbf{d}_1 \in DgSp_{(\omega,\leq)}(X_1)$ and $\mathbf{d}_2 \in DgSp_{(\omega,\leq)}(X_2)$. Then there exist programs that build computable copies $\mathbf{L}_1$ and $\mathbf{L}_2$ of $(\omega, \leq)$ where the image of X_i in $\mathbf{L}_i$ has degree $\mathbf{d}_i$. Using these programs we can build $\mathbf{L}_1$ having the odd numbers as its domain, and $\mathbf{L}_2$ having the even numbers as its domain (again with the image of X_i in $\mathbf{L}_i$ having degree $\mathbf{d}_i$). Finally, we build a computable ordering $\mathbf{L} \cong (\omega, \leq_\zeta)$ as follows:

- Odd numbers $\leq_{\mathbf{L}}$ Even numbers,
- $2n + 1 \leq_{\mathbf{L}} 2m + 1 \Leftrightarrow m \leq_{\mathbf{L}_1} n$, and
- $2n \leq_{\mathbf{L}} 2m \Leftrightarrow n \leq_{\mathbf{L}_2} m$.

Since $deg((X_1)_{\mathbf{L}_1}) = \mathbf{d}_1$ and $deg((X_2)_{\mathbf{L}_2}) = \mathbf{d}_2$, we have $deg(X_{\mathbf{L}}) = deg((X_1)_{\mathbf{L}_1}) \cup deg((X_2)_{\mathbf{L}_2}) = \mathbf{d}_1 \cup \mathbf{d}_2$ and hence $\mathbf{d}_1 \cup \mathbf{d}_2 \in DgSp_{(\omega,\leq_\zeta)}(X)$.

To prove the reverse inclusion, let $\mathbf{d} \in DgSp_{(\omega,\leq_\zeta)}(X)$ and let $\mathbf{L}$ be a computable presentation of $(\omega, \leq_\zeta)$ with $deg(X_{\mathbf{L}}) = \mathbf{d}$. We claim that we can build a copy $\tilde{\mathbf{L}}$ of $(\omega, \leq_\zeta)$ that is computably isomorphic to $\mathbf{L}$ where all odd numbers in ω are $\leq_{\tilde{\mathbf{L}}}$-below the even numbers. Assume that we know $0_{\mathbf{L}} = x_0$, i.e. x_0 is the "zero" in $\mathbf{L}$. We will run the enumeration of $\mathbf{L}$ and build our computable isomorphism φ in stages as follows:

When the number s appears at stage s, we will compute whether $s <_{\mathbf{L}} x_0$ or $s \geq_{\mathbf{L}} x_0$. If $s <_{\mathbf{L}} x_0$ then we define $\varphi(s)$ to be the smallest odd number not yet in the domain of $\tilde{\mathbf{L}}[s]$; if $s \geq_{\mathbf{L}} x_0$ then we define $\varphi(s)$ to be the smallest even number not yet in $\tilde{\mathbf{L}}[s]$. If a is the current predecessor and b is the current successor of s in $\mathbf{L}[s]$ then we declare $\varphi(a) \leq_{\tilde{\mathbf{L}}} \varphi(s) \leq_{\tilde{\mathbf{L}}} \varphi(b)$ in $\tilde{\mathbf{L}}$.

It is clear, by our contruction, that all of the odd numbers in appear $\tilde{\mathbf{L}}$-below all of the even numbers in $\tilde{\mathbf{L}}$ and since $\mathbf{L}$ is infinite in both directions, $\tilde{\mathbf{L}}$ will have domain ω. Since we define the value of $\varphi(x)$ at the stage when x first appears in $\mathbf{L}$, φ is in fact a computable isomorphism. Let $y, z \in \tilde{\mathbf{L}}$. To compute whether $y \leq_{\tilde{\mathbf{L}}} z$ we need only wait until they are both placed in the $\tilde{\mathbf{L}}$-ordering, which must occur after finitely

many stages. (Of course, another way is to determine $\varphi^{-1}(y)$ and $\varphi^{-1}(z)$ and compute whether $\varphi^{-1}(y) \leq_{\mathbf{L}} \varphi^{-1}(z)$.) So $\tilde{\mathbf{L}}$ is indeed a computable copy of $(\zeta, \leq)$ with the desired property.

Since $\mathbf{L} \cong \tilde{\mathbf{L}}$ via a computable isomorphism, we have $deg(X_{\tilde{\mathbf{L}}}) = deg(X_{\mathbf{L}}) = \mathbf{d}$. Since $\tilde{\mathbf{L}}_{\geq 0_{\tilde{\mathbf{L}}}} \cong (\omega, \leq)$ and $\tilde{\mathbf{L}}_{<0_{\tilde{\mathbf{L}}}} \cong (\omega^*, \leq^*)$ we can write the degree of $X_{\tilde{\mathbf{L}}}$ as $\mathbf{d}_1 \cup \mathbf{d}_2$ where $\mathbf{d}_1$ is the degree of $X_{\tilde{\mathbf{L}}} \cap \{odd\ numbers\}$ and d_2 is the degree of $X_{\tilde{\mathbf{L}}} \cap \{even\ numbers\}$. So $\mathbf{d} = \mathbf{d}_1 \cup \mathbf{d}_2$ where $\mathbf{d}_i \in DgSp_{(\omega,\leq)}(X_i)$ and hence $DgSp_{(\zeta,\leq)}(X) \subseteq DgSp_{(\omega,\leq)}(X_1) \oplus DgSp_{(\omega,\leq)}(X_2)$ as desired. □

With this result in mind, we will now dicuss how our results from Chapter 2 extend to the integers.

First, we recall Theorem 2.4 for unary relations on the structure $(\omega, \leq)$. We now wish to prove an analogous result for the integers. Since 2.4 is about the existence of a high c.e. set X that is not order-computable, we get the following Corollary from our discussion above.

Corollary 3.2. *Let $\boldsymbol{d}$ be a high c.e. degree. Then there exists a unary relation X on $(\omega, \leq_\zeta)$ such that $deg(X) = \boldsymbol{d}$ and $\boldsymbol{0} \notin DgSp_{(\omega,\leq_\zeta)}(X)$.*

Proof. Fix a high c.e. degree $\mathbf{d}$. Then by Theorem 2.4, there exists a unary relation X on $(\omega, \leq)$ such that $deg(X) = \mathbf{d}$ and $\mathbf{0} \notin DgSp_{(\omega,\leq)}(X)$. Since $X \subset \mathbb{N}$, we have $DgSp_{(\omega,\leq_\zeta)}(X) = DgSp_{(\omega,\leq)}(X)$ and hence $\mathbf{0} \notin DgSp_{(\zeta,\leq)}(X)$ as desired. □

Next we will examine how Theorems 2.1 and 2.2 extend to the integers using Proposition 3.1.

Corollary 3.3. *Let X be a unary relation on $(\omega, \leq_\zeta)$. Suppose that X forms a c.e. set such that either $X \cap \{2n+1 : n \in \mathbb{N}\}$ is infinite and coinfinite within the odd numbers, or $X \cap \{2n : n \in \mathbb{N}\}$ is infinite and coinfinite within the even numbers (or both). Let Y be a c.e. set such that $X \leq_T Y$. Then there is a computable presentation $\boldsymbol{L}$ of $(\omega, \leq_\zeta)$ such that $X_{\boldsymbol{L}} \equiv_T Y$.*

Proof. Let Y be c.e. set such that $X \leq_T Y$. To prove the corollary, we need to show that $deg(Y) \in DgSp_{(\zeta,\leq)}(X)$. Let $X = X_1 \oplus X_2$, as defined in Proposition 3.1. Since $X \leq_T Y$, we have $X_1, X_2 \leq_T Y$ as well (by a property of the join [5]). By assumption, at least one of X_1 and X_2 is infinite and coinfinite. Let us suppose, without loss of generality that it is X_2. We can consider $X_2 \leq_T Y$ as a unary relation on $(\omega, \leq)$, and hence by Theorem 2.1, $deg(Y) \in DgSp_{(\omega,\leq)}(X_2)$. By Proposition 3.1, $deg(X_1) \cup deg(Y) \in DgSp_{(\omega,\leq_\zeta)}(X)$. But $X_1 \leq_T Y \Rightarrow deg(X_1) \cup deg(Y) = deg(Y)$, so we have proven the result. □

Theorem 2.2 extends to the integers on a similar manner as 2.1.

Corollary 3.4. *Let X be a unary relation on $(\omega, \leq_\zeta)$ and suppose that there exists a computable copy $\boldsymbol{L}$ of $(\omega, \leq_\zeta)$ such that $X_{\boldsymbol{L}}$ is computable. If $X \cap \{2n+1 : n \in \mathbb{N}\}$ is infinite and coinfinite within the odd numbers, or $X \cap \{2n : n \in \mathbb{N}\}$ is infinite and coinfinite within the even numbers (or both), then $DgSp_{(\omega,\leq_\zeta)}(X)$ contains all Δ_2 degrees.*

Proof. Again, let $X = X_1 \oplus X_2$ as defined in Proposition 3.1. Since $\mathbf{0} \in DgSp_{(\omega,\leq_\zeta)}(X)$, we have $\mathbf{0} \in DgSp_{(\omega,\leq)}(X_1)$ and $\mathbf{0} \in DgSp_{(\omega,\leq)}(X_2)$. We know that X is infinite and coinfinite on at least one infinite segment, so we know where we can find our infinitely many bit alternations in $X_{\mathbf{L}}$. This time, let us assume that $X_{\mathbf{L}}$ is infinite and coinfinite to the left of 0. Then X_1 is infinite and coinfinite, and thus by Theorem 2.2, we know that $DgSp_{(\omega,\leq)}(X_1)$ contains all Δ_2 degrees. Since $\mathbf{d} \cup \mathbf{0} = \mathbf{d}$ for any degree $\mathbf{d}$, then by Proposition 3.1, $DgSp_{(\omega,\leq_\zeta)}(X)$ contains all Δ_2 degrees. □

Finally, in Theorem 2.3 we stated that a sufficient condition for a set X to be order-computable is for X to be of non-high degree. We will now extend the notion of order-computable sets to unary relations on the structure $(\omega, \leq_\zeta)$. Again, we need to examine how the given set $X \subseteq \mathbb{Z}$ is distributed across the the two segments of $(\omega, \leq_\zeta)$.

Corollary 3.5 *Let X be a unary relation on $(\omega, \leq_\zeta)$. If X is c.e. set whose Turing degree is not high, then $\boldsymbol{0} \in DgSp_{(\omega,\leq_\zeta)}(X)$.*

Proof. Let $X = X_1 \oplus X_2$ as defined in Proposition 3.1.

First, we claim that if X is not high, then neither X_1 nor X_2 can be high. Suppose that $X_1' \geq \emptyset''$. By properties of the join from [5] and [1], we know that $X_1 \leq_T X_1 \oplus X_2$. This implies that $X_1' \leq_T (X_1 \oplus X_2)'$, by a property of the jump [5]. So we have $\emptyset'' \leq_T X_1' \leq_T (X_1 \oplus X_2)'$ and thus $X_1 \oplus X_2 = X$ must be high. This contradiction proves our claim.

Since X_1 and X_2 are not high, then by Theorem 2.3, $\mathbf{0} \in DgSp_{(\omega,\leq)}(X_1)$ and $\mathbf{0} \in DgSp_{(\omega,\leq)}(X_2)$. Therefore $\mathbf{0} = \mathbf{0} \cup \mathbf{0} \in DgSp_{(\omega,\leq_\zeta)}(X)$ by Proposition 3.1. □

We see that each analogous result follows quite directly from the correspondence in Proposition 3.1. We are now left to ask whether a set of Turing degrees can be obtained by considering $DgSp_{(\omega,\leq)}(X_1) \oplus DgSp_{(\omega,\leq)}(X_2)$ for $X_1, X_2 \subseteq \mathbb{N}$, but cannot be realized as the degree spectra of a single unary relation X on $(\omega, \leq)$.

References

[1] S. Barry Cooper. *Computability theory.* Chapman & Hall/CRC, Boca Raton, FL, 2004.

[2] R. Downey, B. Khoussainov, J. Miller, and L. Yu. Degree spectra of unary relations on $(\omega, \leq)$. to appear.

[3] Denis Hirschfeldt, Russell Miller, and Sergei Podzorov. Order-computable sets. *Notre Dame J. Formal Logic*, 48(3):317–347 (electronic), 2007.

[4] Denis R. Hirschfeldt. Degree spectra of relations on computable structures. *Bull. Symbolic Logic*, 6(2):197–212, 2000.

[5] 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.