MARTIN'S CONJECTURE FOR REGRESSIVE FUNCTIONS ON THE HYPERARITHMETIC DEGREES

PATRICK LUTZ

Abstract. We answer a question of Slaman and Steel by showing that a version of Martin's conjecture holds for all regressive functions on the hyperarithmetic degrees. A key step in our proof, which may have applications to other cases of Martin's conjecture, consists of showing that we can always reduce to the case of a continuous function.

1. Introduction

Martin's conjecture is an ambitious attempt to classify the limit behavior of functions on the Turing degrees under strong set theoretic hypotheses (namely the axiom of determinacy). The full conjecture is still open, but several special cases have been proved. In particular, in [SS88], Slaman and Steel used an ingenious coding argument to prove that Martin's conjecture holds for all "regressive" functions on the Turing degrees.

Theorem 1.1 ($\mathsf{ZF} + \mathsf{AD}$; Slaman and Steel)**.** *If $f\colon 2^\omega \to 2^\omega$ is a Turing-invariant function such that $f(x) \leq_T x$ for all x then either f is constant on a cone or $f(x) \equiv_T x$ on a cone.*

They also asked whether the analogous theorem for hyperarithmetic reducibility holds. In other words, is it possible to prove a version of Martin's conjecture for regressive functions on the hyperarithmetic degrees? Their motivation was as follows. A regressive function on the Turing degrees can be written as a countable union of continuous functions, and their coding argument relied strongly on the properties of continuous functions. A regressive function on the hyperarithmetic degrees, however, can only be written as a countable union of Borel functions and thus forms a natural test case to see whether their argument can be extended to deal with non-continuous functions.

The main result of this paper is to answer their question in the affirmative. Namely we will prove the following theorem.

Theorem 1.2 ($\mathsf{ZF} + \mathsf{AD}$)**.** *Let $f\colon 2^\omega \to 2^\omega$ be a hyp-invariant function such that $f(x) \leq_H x$ for all x. Then either f is constant on a cone of hyperarithmetic degrees or $f(x) \equiv_H x$ on a cone of hyperarithmetic degrees.*

There are a few interesting things to note about our proof. First, instead of adapting Slaman and Steel's methods to work with non-continuous functions, we instead show that f—despite potentially being far from continuous—can be replaced by a hyp-equivalent function which *is* continuous. We still have to replace Slaman and Steel's coding argument with a new coding argument that works with hyperarithmetic reducibility rather than Turing reducibility, but in doing so we make heavy use of the fact that we can assume we are dealing with a continuous function.

This suggests that in some cases of Martin's conjecture where the functions being considered are not continuous, it may still be possible to replace them with related functions which are continuous. This idea has already borne fruit in the form of [LS20], where it is combined with

a refined version of the coding arguments introduced in this paper to prove part 1 of Martin's conjecture for order-preserving functions.

Second, our results cast at least a little doubt on the idea that any use of determinacy in proving Martin's conjecture will be "local" (that is, the idea that only Borel determinacy is needed when dealing with Borel functions, and so on). Our proof seems to use more than Borel determinacy, even when the functions being considered are assumed to be Borel (specifically, our proof uses analytic determinacy). In section 4, we show that Borel determinacy *is* sufficient, but this requires a more careful analysis that was not needed for the AD proof.

Third, our reduction to the case of a continuous function is quite flexible and seems to work in many different degree structures, including the arithmetic degrees. Somewhat surprisingly, it seems much harder to adapt the coding arguments used by Slaman and Steel, even once we are allowed to assume we are dealing with a continuous function. In this paper, we have to use a different coding argument than the one used by Slaman and Steel, and in doing so we have to rely on the Σ^1_1-bounding theorem. Also, we have so far not been able to modify either coding argument to work for arithmetic reducibility and, in our opinion, the regressive case of Martin's conjecture on the arithmetic degrees is an interesting open question.

2. Preliminaries

In this section we will provide some background on hyperarithmetic reducibility and on Martin's conjecture and then state some lemmas that we will use in the proof of theorem 1.2. All the lemmas are standard, with the exception of lemma 2.8, which we will see has a simple proof using standard techniques. For the reader intimidated by the axiom of determinacy, we note that the only way we will use determinacy in this paper is in the form of lemma 2.8.

2.1. **Background on Hyperarithmetic Reducibility.** The easiest definition of hyperarithmetic reducibility is that $y \leq_H x$ if y is $\Delta^1_1(x)$ definable (in which case we will often say that y is hyperarithmetic in x). It is not very hard to see that this relation is transitive and thus deserves the title "reducibility." As usual, we can then define hyperarithmetic equivalence and the structure of the hyperarithmetic degrees.

But there is another characterization of hyperarithmetic reducibility which is often useful and which we will now explain. Letting ω_1^x denote the least countable ordinal with no presentation computable from x, Davis observed that there is a well-defined (up to many-one degree) notion of iterating the jump of x up to any ordinal below ω_1^x. If $\alpha < \omega_1^x$ then we denote this α^{th} jump of x by $x^{(\alpha)}$. It is possible to show that y is hyperarithmetic in x if and only if y is computable from $x^{(\alpha)}$ for some $\alpha < \omega_1^x$. In fact, this was Davis's original definition of hyperarithmetic reducibility—its equivalence to the definition involving Δ^1_1 definability was only shown several years later by Kleene (and by the way, this equivalence is not at all obvious—it is maybe surprising that the definition in terms of transfinite iterates of the jump is even transitive since it is not obvious that $x \leq_H y$ implies that $\omega_1^x \leq \omega_1^y$).

It will be helpful later in the paper if we make some of this more precise. Suppose r is a linear order on $\mathbb{N}$ and 0 is the minimum element according to r. If x is any real, then a jump hierarchy on r which starts with x is a set $H \subset \mathbb{N}^2$ such that the 0^{th} column of H is x and for each $n \neq 0$, the n^{th} column of H is equal to the jump of the smaller columns of H (smaller according to the ordering given by r). In other words, if we define

$$H_n = \{i \mid \langle n, i\rangle \in H\}$$
$$H_{<n} = \{\langle m, i\rangle \mid m <_r n \text{ and } \langle m, i\rangle \in H\}$$

then we have $H_0 = x$ and $H_n = H'_{<n}$ for all $n \neq 0$.

If r is a presentation of a well-order then there is always a unique H satisfying the conditions above. Moreover, if $\alpha < \omega_1^x$ and r is a presentation of α computable from x then the many-one degree of the unique jump hierarchy on r starting from x is independent of the choice of r. Such a jump hierarchy is considered to be the α^{th} jump of x (which is only well-defined up to many-one degree). This makes precise the alternative characterization of hyperarithmetic reducibility mentioned above.

It is also worth mentioning here that hyperarithmetic reducibility is closely connected to Borel measurability. Just as continuous functions correspond to computable functions (relative to some oracle), Borel functions correspond to hyperarithmetic functions (relative to some oracle). More precisely, if f is Borel then there is some countable ordinal α, some presentation r of α, some real y and some Turing functional Φ such that for all x, $f(x) = \Phi((x \oplus y)^{(\alpha)})$, where $(x \oplus y)^{(\alpha)}$ is taken to mean the unique jump hierarchy on r starting from $x \oplus y$.

2.2. **Background on Martin's Conjecture.** As mentioned in the introduction, Martin's conjecture is an attempt to classify the limit behavior of functions on the Turing degrees under strong set theoretic hypotheses. It is traditionally divided into two parts. We will only discuss the first part here, since that is all that is relevant for this paper.

Very roughly, part 1 of Martin's conjecture states that if f is a function from the Turing degrees to the Turing degrees then either $f(x)$ is constant for all large enough x or $f(x) \geq_T x$ for all large enough x. There are three things to explain here. First, a caveat: the conjecture is actually stated not in terms of functions on the Turing degrees, but in terms of Turing invariant functions on the reals. Second, we need to state precisely what "for all large enough x" really means. Third, the conjecture is false in ZFC and is instead stated as a conjecture in the theory $\mathsf{ZF} + \mathsf{AD}$ (or sometimes $\mathsf{ZF} + \mathsf{AD} + \mathsf{DC}_{\mathbb{R}}$, though we will not need to use $\mathsf{DC}_{\mathbb{R}}$ in this paper). We will now explain each of these points in more detail.

First, let's define precisely what we mean by a Turing invariant function on the reals. A function $f\colon 2^\omega \to 2^\omega$ is called Turing invariant if for all x and y in 2^ω,

$$x \equiv_T y \implies f(x) \equiv_T f(y).$$

The point is that a Turing invariant function f induces a function on the Turing degrees. Using the axiom of choice, it is clear that every function on the Turing degrees arises from a Turing invariant function on the reals, but this may fail in ZF (though it is true again if we assume $\mathsf{AD}_{\mathbb{R}}$, a strengthening of the axiom of determinacy). So Martin's conjecture is actually only classifying the behavior of functions on the Turing degrees which come from Turing invariant functions on the reals.

Since it will be useful to us, we will also mention here the definition of a Turing invariant set of reals. A subset $A \subseteq 2^\omega$ is called Turing invariant if for all x and y in 2^ω,

$$x \equiv_T y \implies (x \in A \leftrightarrow y \in A).$$

Next, let's explain what we mean by "all large enough x." The key concept is that of a cone of Turing degrees (which is actually a Turing invariant subset of 2^ω rather than a subset of the Turing degrees): a cone of Turing degrees is a set of the form $\{x \in 2^\omega \mid x \geq_T y\}$ for some fixed y. This y is called the base of the cone and the cone is sometimes referred to as the cone above y. What we mean by "all large enough x" is simply "for all x in some cone."

Third, we will mention a few things about the axiom of determinacy. The axiom of determinacy (often written AD) is a strong axiom of set theory which is inconsistent with the axiom of choice and whose consistency strength is known (it is equiconsistent with the existence of infinitely many Woodin cardinals, in case you are curious). We will not give a

definition of the axiom of determinacy here, but simply mention the following fact, which is one of the main consequences of AD for computability theory.

Theorem 2.1 ($\mathsf{ZF} + \mathsf{AD}$; Martin)**.** *If A is a Turing invariant subset of 2^ω then either A contains a cone or A is disjoint from a cone.*

By the way, there is a weak form of determinacy called "Borel determinacy" which *is* provable in ZF and which is enough to prove a theorem 2.1 if the set A is assumed to be Borel.

We can now give a formal statement of part 1 of Martin's conjecture.

Conjecture 2.2 (Part 1 of Martin's Conjecture)**.** *Assuming $\mathsf{ZF} + \mathsf{AD}$, if $f\colon 2^\omega \to 2^\omega$ is a Turing invariant function then either $f(x) \geq_T x$ for all x in some cone or there is some y such that $f(x) \equiv_T y$ for all x in some cone.*

In a slight abuse of terminology, the latter possibility in the conjecture is often written as "f is constant on a cone" (even though it is the quotient of f by Turing equivalence which is constant, rather than f itself).

Finally, we mention that for many degree structures besides the Turing degrees (and in particular for the hyperarithmetic degrees), it is possible to state a sensible version of Martin's conjecture by just swapping out Turing reducibility for the appropriate alternative notion of reducibility in the definitions of "Turing invariant function" and "cone of Turing degrees." This is reasonable to do in part because theorem 2.1 works for pretty much any notion of reducibility stronger than Turing reducibility (and also for many which are weaker).

2.3. **Determinacy Lemmas.** We now state a few lemmas that will help us apply determinacy even in situations where we have to deal with non-Turing invariant sets of reals. The key notion is that of a "pointed perfect tree."

Definition 2.3. *A perfect tree, T, is a tree such that every node in T has a pair of incompatible extensions which are both in T.*

Definition 2.4. *A pointed perfect tree, T, is a perfect tree such that every path through T computes T.*

Notation 2.5. If T is a tree, we will use $[T]$ to refer to the set of paths through T.

The reason pointed perfect trees are useful to work with is that if T is a pointed perfect tree then $[T]$ contains a representative of every Turing degree which is above the Turing degree of T. Next we will see that determinacy can be used to get pointed perfect trees. For a proof of lemma 2.7, see [MSS16], lemma 3.5.

Definition 2.6. *A set $A \subseteq 2^\omega$ is cofinal in the Turing degrees if for all x there is some $y \geq_T x$ such that $y \in A$ (note that A is not required to be Turing invariant).*

Lemma 2.7 ($\mathsf{ZF} + \mathsf{AD}$)**.** *Suppose $A \subseteq 2^\omega$ is cofinal in the Turing degrees and h is a function on A with countable range. Then there is a pointed perfect tree on which h is constant.*

The following lemma will be our only use of determinacy in the proofs in the rest of this paper. Essentially it is a kind of computable uniformization principle provable from AD.

Lemma 2.8 ($\mathsf{ZF} + \mathsf{AD}$)**.** *Suppose R is a binary relation on 2^ω such that*

- *The domain of R is cofinal in the Turing degrees: for all z there is some $x \geq_T z$ and some y such that $(x, y) \in R$*
- *and R is a subset of Turing reducibility: for every $(x, y) \in R$, $x \geq_T y$.*

Then there is a pointed perfect tree T and a Turing functional Φ such that for every $x \in [T]$, $\Phi(x)$ is total and $(x, \Phi(x)) \in R$. In other words, Φ is a computable choice function for R on $[T]$.

Proof. For each x in the domain of R there is some e such that $\Phi_e(x)$ is total and $R(x, \Phi_e(x))$ holds. Let e_x denote the smallest such e. By determinacy (in the form of lemma 2.7), there is a pointed perfect tree T on which e_x is constant. Let e be this constant value. Then T and Φ_e satisfy the conclusion of the lemma. □

It will be useful below to note that if A and h in lemma 2.7 are Borel then the result is provable in ZF, and similarly that if R in the above lemma is assumed to be Borel, then that result, too, is provable in ZF.

2.4. **Pointed Perfect Tree Lemmas.** Now we will state a couple of lemmas that are helpful when working with pointed perfect trees. These lemmas do not require the axiom of determinacy. The first lemma can be proved using the same kind of arguments as in Spector's construction of a minimal degree and the second is a routine application of compactness.

Lemma 2.9. *Suppose T is a pointed perfect tree and Φ is a Turing functional such that $\Phi(x)$ is total for every $x \in [T]$. Then either Φ is constant on a pointed perfect subtree of T or Φ is injective on a pointed perfect subtree of T.*

Lemma 2.10. *Suppose T is a perfect tree and Φ is a Turing functional such that $\Phi(x)$ is total for every $x \in [T]$ and Φ is injective on $[T]$. Then for each $x \in [T]$,*

$$\Phi(x) \oplus T \geq_T x.$$

In fact, this reduction is even uniform in x (though we won't need to use that fact in this paper).

2.5. **Computable Linear Orders.** To work with hyperarithmetic reducibility, we will need to make use of a few facts about computable linear orders and computable well-orders. Proofs can be found in [Sac90].

One of the most important facts about computable linear orders is the Σ^1_1-bounding theorem. Essentially it says that every Σ^1_1-definable collection of well-orders is bounded below a computable ordinal. The theorem comes in multiple flavors, depending on whether we are talking about sets of programs which compute presentations of linear orders, or real numbers which *are* presentations of linear orders and depending on whether the Σ^1_1 definition is boldface, lightface, or lightface relative to some fixed real. Below, we just state the two versions that we will need in this paper.

Theorem 2.11. *Suppose that x is a real and A is a $\Sigma^1_1(x)$ definable set of codes for programs such that for every e in A, $\Phi_e(x)$ is a presentation of a well-order. Then there is some $\alpha < \omega_1^x$ which is greater than every ordinal with a presentation coded by an element of A.*

Theorem 2.12. *If A is a $\mathbf{\Sigma^1_1}$ definable set of presentations of well-orders then there is some $\alpha < \omega_1$ which is greater than every ordinal with a presentation in A.*

We will also need some ideas originally introduced by Harrison in [Har68].

Definition 2.13. *If x is a real and r is a real computable from x that codes a presentation of a linear order, then r is a pseudo-well-order relative to x if it is ill-founded but contains no infinite descending sequence which is hyperarithmetic in x.*

Lemma 2.14. *If r is a presentation of a linear order that is computable from a real x then the assertion "r has no infinite descending sequence which is hyperarithmetic in x" is equivalent to a $\Sigma^1_1(x)$ formula.*

Lemma 2.15. *If r is a pseudo-well-order relative to x and H is a jump hierarchy on r that starts with x then H computes every real which is hyperarithmetic in x.*

3. Proof of the Main Theorem

In this section, we will prove theorem 1.2. Before we launch into the details of the proof, we will give an outline of the general strategy. And before we do that, we will recall the general strategy followed by Slaman and Steel in their proof of theorem 1.1. The steps of their proof are essentially as follows.

- First, use determinacy to show that there is a pointed perfect tree on which f is computable. Then use lemma 2.9 to show that we can also assume f is injective.
- Next, show that if x is in the pointed perfect tree, then every function computable from x is dominated by a function computable from $f(x)$. The idea is that if x computes a function which is not dominated by any function computable from $f(x)$ then x can diagonalize against $f(x)$ by using this function to guess convergence times for $f(x)$ programs. The diagonalization produces a real y in the same Turing degree as x such that $f(x)$ cannot compute $f(y)$, thereby contradicting the Turing invariance of f.
- Once you can assume that f is computable and injective on a pointed perfect tree and that if x is in this tree then every function computed by x is dominated by a function computed by $f(x)$, use a coding argument to show that $f(x) \geq_T x$. The coding argument works by coding bits of x into the relative growth rates of two fast growing functions computed by $f(x)$.

Our proof makes three main modifications to this outline. First, instead of showing that f is computable on some pointed perfect tree, we show that f is hyp-equivalent to some computable function on a pointed perfect tree. Thus we may work with that function instead of f. Second, instead of showing that every fast growing function computed by x is dominated by a function computed by $f(x)$, we show that every well-order computed by x embeds into a well-order computed by $f(x)$—in other words that $\omega_1^x = \omega_1^{f(x)}$. Third, instead of coding bits of x into the relative growth rates of fast growing functions computed by $f(x)$, we code the bits of x into the Kolmogorov complexities of initial segments of reals computed by $f(x)$ (though it is not necessary to know anything about Kolmogorov complexity to follow our argument). Also, to be able to carry out the coding argument, we will first have to use a trick involving Σ^1_1-bounding. To sum up, here's an outline of our proof.

- First, we will use determinacy to replace f with a hyp-equivalent function which is computable on a pointed perfect tree. By using lemma 2.9, we can also assume that f is injective.
- Next we show that $\omega_1^{f(x)} = \omega_1^x$ for all x in the pointed perfect tree. The idea is if $\omega_1^{f(x)}$ was less than ω_1^x then x would be able to diagonalize against $f(x)$ by using $\omega_1^{f(x)}$ jumps.
- Once we are able to assume that f is computable and injective and that $\omega_1^{f(x)} = \omega_1^x$, we will use a coding argument to show that $f(x) \geq_H x$. In our coding argument, it will be important to know that there is a single ordinal $\alpha < \omega_1^x$ such that for every real y in the same Turing degree as x, $f(x)^{(\alpha)}$ computes $f(y)$. We will prove this fact using Σ^1_1-bounding.

And now it's time to present the actual proof.

3.1. Replacing f with an injective, computable function.

First we will show that f can be replaced by a computable function. This is the only part of the proof that uses determinacy.

Lemma 3.1 (ZF + AD)**.** *Suppose $f\colon 2^\omega \to 2^\omega$ is hyp-invariant and hyp-regressive. Then there is a Turing functional Φ and a pointed perfect tree T such that for all $x \in [T]$, $\Phi(x)$ is total and $\Phi(x) \equiv_H f(x)$.*

Proof. Consider the following binary relation, R.

$$R(x,z) \iff x \geq_T z \text{ and } \exists y\,(y \equiv_H x \wedge f(y) = z).$$

So basically R just says that x computes z and z is in the image of f on the hyperdegree of x. I claim that the domain of R is cofinal in the Turing degrees and thus lemma 2.8 applies. To prove this, we need to start with an arbitrary x and show that some real which computes x is in the domain of R. So let $x \in 2^\omega$. Since $f(x) \leq_H x$, there is some $\alpha < \omega_1^x$ such that $f(x) \leq_T x^{(\alpha)}$. And this $x^{(\alpha)}$ is the real we are after: it clearly computes x and since $R(x^{(\alpha)}, f(x))$ holds, $x^{(\alpha)}$ is indeed in the domain of R.

So by lemma 2.8, there is a pointed perfect tree T and a Turing functional Φ such that for all $x \in [T]$, $\Phi(x)$ is total and is in the image of f on the hyperdegree of x. Since f is hyp-invariant, this implies that $f(x) \equiv_H \Phi(x)$ for every $x \in [T]$. □

For the rest of the proof we will simply assume that f is computable on a pointed perfect tree. It will also be convenient to assume that f is injective on a pointed perfect tree, which we show next.

Lemma 3.2 (ZF)**.** *Suppose T is a pointed perfect tree and $f\colon 2^\omega \to 2^\omega$ is a hyp-invariant function which is computable on $[T]$. Then either f is constant on a cone of hyperdegrees or f is injective on a pointed perfect subtree of T.*

Proof. By lemma 2.9, either f is constant on a pointed perfect subtree of T or f is injective on a pointed perfect subtree of T. In the former case, f is constant on a cone of hyperdegrees and in the latter case, we are done. □

For the rest of the proof, we will deal with the case of a hyp-invariant function, f, which is computable and injective on a pointed perfect tree, T. We will show that for any x in $[T]$, $f(x) \geq_H x$. There are two cases: when $\omega_1^{f(x)} < \omega_1^x$ and when $\omega_1^{f(x)} = \omega_1^x$. We will show that the first case is impossible and that if we are in the second case then we can complete the proof.

3.2. **Proving that f preserves ω_1^x.** We will now show that for any $x \in [T]$, f preserves ω_1^x. We will do this by deriving a contradiction from the assumption that $\omega_1^{f(x)} < \omega_1^x$. The basic idea is that in this case we can diagonalize against $f(x)$. Namely, we can use $\omega_1^{f(x)}$ jumps of x to compute a real y so that $f(x)$ cannot compute $f(y)$ with fewer than $\omega_1^{f(x)}$ jumps (and hence $f(x)$ cannot be hyp-equivalent to $f(y)$). Since $\omega_1^x > \omega_1^{f(x)}$, this y can be made hyp-equivalent to x, which violates the hyp-invariance of f. We now give the formal proof.

Lemma 3.3 (ZF)**.** *Suppose T is a pointed perfect tree and f is a hyp-invariant function which is computable and injective on $[T]$. Then for every $x \in [T]$, $\omega_1^{f(x)} = \omega_1^x$.*

Proof. Suppose for contradiction that for some $x \in [T]$, $\omega_1^{f(x)} < \omega_1^x$. Let $\alpha = \omega_1^{f(x)}$. The key point is that for every $y \in [T]$ which is hyp-equivalent to x, $x^{(\alpha)}$ computes y.

Why is that? Well, if y is in the same hyperdegree as x then $f(y)$ is in the same hyperdegree as $f(x)$. So by definition of α, there is some $\beta < \alpha$ such that $f(x)^{(\beta)} \geq_T f(y)$. We then have

the following calculation.

$$\begin{aligned} x^{(\alpha)} &\geq_T x^{(\beta)} && \text{because } \beta < \alpha \\ &\geq_T x^{(\beta)} \oplus T && \text{because } T \text{ is pointed} \\ &\geq_T f(x)^{(\beta)} \oplus T && \text{because } f(x) \leq_T x \\ &\geq_T f(y) \oplus T && \text{by definition of } \beta \\ &\geq_T y && \text{by lemma 2.10.} \end{aligned}$$

We can now finish the proof easily. Since T is pointed, we can pick some $y \in [T]$ which is Turing equivalent to $x^{(\alpha+1)}$. Since $\alpha < \omega_1^x$, this y is hyp-equivalent to x. But it obviously is not computable from $x^{(\alpha)}$, so we have reached a contradiction. □

3.3. **Coding argument.** In this part of the proof, we will explain how to code x into an real in the same hyperarithmetic degree as $f(x)$. The argument has some similarity to the proof of a basis theorem for perfect sets given by Groszek and Slaman in [GS98] (which itself has some similarity to the coding argument given in [SS88]). Before giving the coding argument, however, we will first show that for every $x \in [T]$ there is a uniform bound on the number of jumps that $f(x)$ takes to compute $f(y)$ for any $y \in [T]$ which is Turing equivalent to x.

Lemma 3.4 (ZF)**.** *Suppose T is a pointed perfect tree, f is a hyp-invariant function which is computable on $[T]$, and $x \in [T]$. Then there is some $\alpha < \omega_1^x$ such that if $y \in [T]$ is Turing equivalent to x then $f(y) \leq_T f(x)^{(\alpha)}$.*

Proof. The main idea is just to use Σ^1_1-bounding. Let A be the set of codes for linear orders r which are computable from $f(x)$ and such that

- r has no infinite descending sequence which is hyperarithmetic in $f(x)$
- and there is some $y \equiv_T x$ in $[T]$ and some jump hierarchy H on r such that $H_0 = f(x)$ and H does not compute $f(y)$.

By lemma 2.14 (and since $f(x)$ is computable in x), A is $\Sigma^1_1(x)$. I claim that A only contains well-orders.

Suppose instead that A contains an ill-founded order, r. Thus r is a pseudo-well-order relative to $f(x)$. Since r is in A, there must be some $y \equiv_T x$ in $[T]$ and some jump hierarchy on r starting with x which does not compute $f(y)$. And since f is hyp-invariant, we must have $f(x) \equiv_H f(y)$. But by lemma 2.15, any jump hierarchy on r which starts with $f(x)$ computes everything in the hyperdegree of $f(x)$, in particular $f(y)$. This is a contradiction, so A must contain only well-orders.

Since A is $\Sigma^1_1(x)$ and contains only well-orders, Σ^1_1-bounding implies that there is some $\alpha < \omega_1^x$ which bounds every well-order in A. This implies that for every $y \equiv_T x$ in $[T]$, $f(y)$ is computable from $f(x)^{(\alpha+1)}$. □

We now come to the coding argument. It replaces a different coding argument in the proof of Slaman and Steel's theorem on regressive functions on the Turing degrees. That argument coded information into the relative growth rates of two fast-growing functions. This coding argument codes information into the relative Kolmogorov complexities of initial segments of three reals (though the reader does not need to be familiar with Kolmogorov complexity to understand the proof below).

As with many arguments of this sort, the construction below is easier to explain one-on-one in front of a blackboard than to communicate in writing; we hope the reader will forgive us.

Lemma 3.5 (ZF). *Suppose T is a pointed perfect tree and f is a hyp-invariant function which is computable and injective on $[T]$. Then $f(x) \geq_H x$ for all $x \in [T]$.*

Proof. Let $x \in [T]$. Our goal is to show that $f(x) \geq_H x$. By 3.3, we know that $\omega_1^x = \omega_1^{f(x)}$ and we will use this fact in the proof. By thinning T, we may assume that x is the base of T (i.e. T is a pointed perfect tree such that $x \equiv_T T$), and hence that any element of T can compute x. We will use this fact below without further comment.

By lemma 3.4, there is some $\alpha < \omega_1^x$ such that for all $y \in [T]$ in the same Turing degree as x, we have $f(x)^{(\alpha)} \geq_T f(y)$. For the remainder of the proof, we will explain how to find reals $a, b, c \in [T]$ which are hyp-equivalent to x such that $x \leq_T f(x)^{(\alpha+2)} \oplus f(a) \oplus f(b) \oplus f(c)$.

To see why this is sufficient to complete the proof, first note that since f is hyp-invariant, $f(a), f(b)$, and $f(c)$ are all hyp-equivalent to $f(x)$. Next, note that since $\omega_1^{f(x)} = \omega_1^x$, α is less than $\omega_1^{f(x)}$ and thus $f(x)^{(\alpha+2)}$ is also hyp-equivalent to $f(x)$. Therefore $f(x)^{(\alpha+2)} \oplus f(a) \oplus f(b) \oplus f(c)$ is hyp-equivalent to $f(x)$ and so if x is Turing below the former then it is hyp below the latter.

We will build a, b, and c in stages. At each stage we will keep track of the following data (supposing that the current stage is n):

- Initial segments A_n, B_n, and C_n of a, b, and c.
- Reals a_n, b_n, and c_n in $[T]$ and Turing equivalent to x, which A_n, B_n, and C_n, respectively, are initial segments of. Think of a_n, b_n, c_n as the current "targets" for a, b, c.
- Initial segments $\widetilde{A}_n, \widetilde{B}_n$, and $\widetilde{C}_n$ of $f(a), f(b)$, and $f(c)$. These are the longest initial segments of $f(a), f(b)$, and $f(c)$ that can be determined from knowing the initial segments A_n, B_n, and C_n of a, b, and c (recall that f is continuous on T).
- Indices for programs $e_{a,n}, e_{b,n}$, and $e_{c,n}$. Think of these as "guesses" as to which programs compute $f(a_n), f(b_n)$, and $f(c_n)$ from $f(x)^{(\alpha)}$.

At the same time, $f(x)$ will be using $f(a), f(b)$, and $f(c)$ to try to follow along with this construction by keeping track of the initial segments $\widetilde{A}_n, \widetilde{B}_n$, and $\widetilde{C}_n$ and the "guesses" $e_{a,n}, e_{b,n}$, and $e_{c,n}$. On each step of the construction we will update the data to code the next bit of x.

On each step, two of a, b, and c will be used to code the next bit of x and the third will play a "helper" role of coding some information to help $f(x)$ follow along with the construction. Which of a, b, and c is playing this "helper" role will simply rotate between them on each step. So, for instance, a will play the helper role every third step.

We will make sure that at the beginning of step n, the "guess" corresponding to whichever real is playing the helper role on step n is correct. E.g. if a is in the helper role on step n then we will need that $e_{a,n}$ is really the index of a program computing $f(a_n)$ from $f(x)^{(\alpha)}$. We will see that the construction ensures this.

We will code the next bit of x into the relative sizes of the guesses for the two reals which are not playing the helper role. E.g. if a is playing the helper role on step n then we will code the next bit of x into which of $e_{b,n+1}$ and $e_{c,n+1}$ is larger—if $x(n) = 0$ then we will make sure $e_{b,n+1} > e_{c,n+1}$ and if $x(n) = 1$ then we will make sure $e_{b,n+1} < e_{c,n+1}$.

To make things more concrete, let's suppose that we are on step n, a is in the helper role, and the next bit of x is a 0 (so we need to make sure $e_{b,n+1} > e_{c,n+1}$). We can assume that $e_{a,n}$ is correct—i.e. that $\Phi_{e_{a,n}}(f(x)^{(\alpha)}) = a_n$—and we need to make sure that this holds of $e_{b,n+1}$ at the end of this step. Here's what we do.

- The target for c will stay the same—i.e. set $c_{n+1} = c_n$.

- Let $e_{c,n+1}$ be the true guess for $c_n = c_{n+1}$—i.e. the least e such that $\Phi_e(f(x)^{(\alpha)} = f(c_n)$ (we know that such an e must exist because we are assuming c_n is in $[T]$ and Turing equivalent to x and thus $f(c_n)$ is computable from $f(x)^{(\alpha)}$).
- Choose some new target b_{n+1} in $[T]$ and of the same Turing degree as x so that b_{n+1} extends B_n and so that for the least e for which $\Phi_e(f(x)^{(\alpha)}) = f(b_{n+1})$, we have $e > e_{c,n+1}$. We can do this because f is injective on T and there are infinitely many reals in $[T]$ extending B_n which are Turing equivalent to x.
- Let m be a number large enough that $e_{c,n+1}$ is the least e such that $\Phi_e(f(x)^{(\alpha)})$ is total and agrees with the first m bits of $f(c_{n+1})$ and likewise for $e_{b,n+1}$.
- Choose some new target a_{n+1} in $[T]$ of the same Turing degree as x which also agrees with the old initial segment A_n of a but which disagrees with a_n and for which the first place such that $f(a_{n+1})$ disagrees with $f(a_n)$ is greater than m, say m'.
- Let $\widetilde{A}_{n+1} = f(a_{n+1}) \upharpoonright m'$.
- Let A_{n+1} be a long enough initial segment of a_{n+1} to ensure that $f(a) \upharpoonright m' = f(a_{n+1}) \upharpoonright m'$ and thus that the first place at which $f(a_n)$ and $f(a_{n+1})$ disagree is m'.
- Set $\widetilde{B}_{n+1}$ and $\widetilde{C}_{n+1}$ to be $f(b_{n+1}) \upharpoonright m'$ and $f(c_{n+1}) \upharpoonright m'$.
- Let B_{n+1} and C_{n+1} be long enough initial segments of b_{n+1} and c_{n+1} to ensure that $f(b)$ and $f(c)$ agree with the first m' bits of $f(b_{n+1})$ and $f(c_{n+1})$ (recall that f is continuous on $[T]$).

Note that by construction, the guesses $e_{b,n+1}$ and $e_{c,n+1}$ are correct. Now let's describe what's happening from $f(x)$'s perspective.

- First we look at $f(a)$. Since it's a's turn to be the helper, we (as $f(x)$) know we should look for the first place where $f(a)$ disagrees with $\Phi_{e_{a,n}}(f(x)^{(\alpha)})$ (which, recall, agrees with $f(a_n)$). So this allows us to retrieve m'.
- Now look at $f(b) \upharpoonright m'$ and $f(c) \upharpoonright m'$. These are the new $\widetilde{B}_n$ and $\widetilde{C}_n$. Calculate the least e such that $\Phi_e(f(x)^{(\alpha)})$ is total and agrees with $f(b)$ up to m'. This is $e_{b,n+1}$. Do the same thing for c.
- Now we check which of $e_{b,n+1}$ and $e_{c,n+1}$ is bigger. That tells us the next bit of x.
- At this point we have the correct guesses for b and c. We may not have a correct guess for a (or even a guess at all) but that doesn't really matter. The only one for which it is vital we have a correct guess at the beginning of the next step is the one that is going to be in helper mode and this will not be a twice in a row (since helper mode always rotates).

To carry out the entire construction to build a, b, and c, we just need to know x and $f(x)^{(\alpha+2)}$ (to figure out which programs are total). Since x computes $f(x)$ and $\alpha < \omega_1^x$, this means that a, b, and c are hyperarithmetic in x. And since $x \equiv_T T$ and T is pointed, x is computable by a, b, and c and thus they are all in the same hyperdegree. At the same time, all that is required to do the parts "from $f(x)$'s perspective" is $f(x)^{(\alpha+2)}$ (to check which programs are total) along with $f(a), f(b)$, and $f(c)$. Hence $x \leq_T f(x)^{(\alpha+2)} \oplus f(a) \oplus f(b) \oplus f(c)$. □

4. The Case of Borel Functions

It is popular to suppose that any proof of Martin's conjecture will only use determinacy in a "local" way—that is, the proof will still work in ZF when restricted to Borel functions, just by replacing the original uses of AD with analogous uses of Borel determinacy.

In this section, we will see that the main result of this paper does hold in ZF when restricted to Borel functions, but that proving this requires using a trick not present in the AD proof

presented above. The trouble is that even if we only consider Borel functions, the proof of lemma 3.1 appears to require analytic determinacy rather than Borel determinacy. However, this can be avoided by a more careful analysis and an appeal to Σ^1_1-bounding.

Here's the key idea. If f is hyp-regressive then we know that for each x there is some $\alpha < \omega_1^x$ such that $x^{(\alpha)}$ computes $f(x)$. We will use Σ^1_1-bounding to find a single α which works for all x. After this, it will be straightforward to modify the proof of lemma 3.1 to only use Borel determinacy.

In the next lemma we will prove this key point that there is a bound on the number of jumps that x needs to compute $f(x)$. Note that since we are restricting ourselves to Borel functons, we can drop the "hyp-regressive" requirement—every Borel function f is automatically regressive on a cone of hyperarithmetic degrees.

Lemma 4.1 (ZF). *Let $f\colon 2^\omega \to 2^\omega$ be a Borel function. Then there is some $\alpha < \omega_1$ such that for all x on a cone of hyperdegrees, $\alpha < \omega_1^x$ and $x^{(\alpha)} \geq_T f(x)$.*

Proof. As noted above, since f is Borel, $f(x) \leq_H x$ on a cone of hyperdegrees. For the rest of the proof, we will implicitly work on this cone and thus we may assume $f(x) \leq_H x$ for all x.

We start by simply writing down the definition of hyperarithmetic reducibility: for each x, we know that $f(x) \leq_H x$ and hence that there is some $\alpha < \omega_1^x$ such that $x^{(\alpha)}$ computes $f(x)$. Our goal is to show that there is some $\alpha < \omega_1$ which is large enough to work for all x. We will do so by using Σ^1_1-bounding.

Let A be the set of presentations of linear orders r such that for some x,

- x computes r
- r has no infinite descending sequences which are hyperarithmetic in x
- and there is a jump hierarchy H on r such that $H_0 = x$ and H does not compute $f(x)$.

By lemma 2.14 plus the fact that f is Borel, the set A is $\mathbf{\Sigma^1_1}$ definable (note that this is boldface rather than lightface because f is Borel but not necessarily lightface Δ^1_1).

Next, I claim that A only contains well-orders. Suppose not and that A contains an ill-founded order, r. Let x witness that r is in A. Then r is a pseudo-well-order relative to x. But by lemma 2.15, this means that any jump hierarchy on r starting with x computes everything hyperarithmetic in x, and in particular, computes $f(x)$. This contradicts the definition of A.

Since A is $\mathbf{\Sigma^1_1}$ and contains only well-orders, $\mathbf{\Sigma^1_1}$-bounding implies that there is some $\alpha < \omega_1$ which bounds everything in A. By the definition of A this means that for every x either $\omega_1^x \leq \alpha$ or $x^{(\alpha+1)} \geq_T f(x)$. And if we go to a cone on which everything computes a presentation of α then we obtain the conclusion of the lemma. □

We can now prove the Borel version of theorem 1.2.

Theorem 4.2 (ZF). *Let $f\colon 2^\omega \to 2^\omega$ be a hyp-invariant Borel function. Then either f is constant on a cone of hyperdegrees or $f(x) \geq_H x$ on a cone of hyperdegrees.*

Proof. By the previous lemma, we can assume there is some $\alpha < \omega_1$ such that for all x on a cone of hyperdegrees, $\alpha < \omega_1^x$ and $x^{(\alpha)} \geq_T f(x)$. Let a be the base of such a cone and let r be a presentation of α computable from a. For the rest of the proof, we will work on the cone above a and we will interpret $x^{(\alpha)}$ to mean the unique jump hierarchy on r that starts with x.

The main idea of the proof is to go through the proof of theorem 1.2 and make sure that every time that proof used determinacy, we can actually get by with just Borel determinacy. The only part of that proof in which we used determinacy was in the proof of lemma 3.1. In particular, we used determinacy by applying lemma 2.8 to the binary relation R defined by

$$R(x,z) \iff x \geq_T z \text{ and } \exists y\,(y \equiv_H x \wedge f(y) = z).$$

The problem is that even if f is Borel, this relation is not $\mathbf{\Delta^1_1}$, but only $\mathbf{\Pi^1_1}$. We will remedy this problem by showing that the relation R can be replaced by the relation S defined by

$$S(x,z) \iff x \geq_T z \text{ and } \exists y \leq_T x\, (y \geq_T a \wedge x \leq_T y^{(\alpha)} \wedge f(y) = z).$$

In particular, we will show that the domain of S is cofinal in the Turing degrees. The requirement that y must compute a is necessary to ensure that $y^{(\alpha)}$ is well-defined (and note that it implies that x must also compute a).

Why is this sufficient? Let's first assume that we can show that the domain of S is cofinal and see why that is enough to complete the proof. Since the definition of S is $\mathbf{\Delta^1_1}$, and satisfies the conditions of lemma 2.8, there is a pointed perfect tree T and a Turing functional Φ such that for all $x \in [T]$, $S(x, \Phi(x))$ holds. Furthermore, the definition of S and the hyp-invariance of f guarantee that $\Phi(x) \equiv_H f(x)$ for all $x \in [T]$. Thus we have recovered the conclusion of lemma 3.1 and the rest of the proof works unchanged.

Why is this true? Now we will show that S has cofinal domain. The proof is very similar to the proof of lemma 3.1. Let x be any real. By joining with a if necessary, we may assume that x is in the cone above a. Since x is in the cone above a, we know that $f(x) \leq_T x^{(\alpha)}$ and so $S(x^{(\alpha)}, f(x))$ holds. Since $x \leq_T x^{(\alpha)}$, we have succeeded in finding something in the domain of S which is above x. □

References

[GS98] Marcia J. Groszek and Theodore A. Slaman. A basis theorem for perfect sets. *The Bulletin of Symbolic Logic*, 4(2):204–209, 1998.

[Har68] Joseph Harrison. Recursive pseudo-well-orderings. *Transactions of the American Mathematical Society*, 131(2):526–543, 1968.

[LS20] Patrick Lutz and Benjamin Siskind. Part 1 of martin's conjecture for order-preserving functions. In preparation, 2020.

[MSS16] Andrew Marks, Theodore A. Slaman, and John R. Steel. *Martin's conjecture, arithmetic equivalence, and countable Borel equivalence relations*, volume 3 of *Lecture Notes in Logic*, pages 493–520. Cambridge University Press, 2016.

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

[SS88] Theodore A. Slaman and John R. Steel. Definable functions on degrees. In Alexander S. Kechris, Donald A. Martin, and John R. Steel, editors, *Cabal Seminar 81–85*, pages 37–55, Berlin, Heidelberg, 1988. Springer Berlin Heidelberg.

Department of Mathematics, University of California, Berkeley
Email address: pglutz@berkeley.edu