January 14, 2011

STRONG JUMP-TRACEABILITY II : K-TRIVIALITY

ROD DOWNEY AND NOAM GREENBERG

Abstract. We show that every strongly jump-traceable set is K-trivial. Unlike other results, we do not assume that the sets in question are computably enumerable.

1. Introduction

The last decade has seen an explosion of research connecting fundamental notions of algorithmic randomness to computational power. One of the most remarkable series of results concerns classes of reals which are close to being computable, as by various measures they have very low complexity. The main example is the notion of K-triviality, which originates in the work of Solovay [25], and was more recently developed starting with Downey, Hirschfeldt, Nies and Stephan [8]. Although this notion is defined in terms of initial-segment complexity (A is K-trivial if for all n, $K(A\upharpoonright_n) \leqslant^+ K(n)$[1]), the celebrated work of Nies, Hirschfeldt and others shows that K triviality coincides with notions such as lowness for K, lowness for Martin-Löf randomness, lowness for weak 2-randomness, and being a base for randomness. All of these notions express feebleness of an oracle with respect to some notion of algorithmic randomness: for example, a real A is low for Martin-Löf randomness if every real which is Martin-Löf random remains Martin-Löf random relative to A; in other words, A cannot detect patterns in Martin-Löf random sets. We refer the reader to [6, 7, 20, 21, 23] for details of such results.

Terwijn [26], and then Terwijn and Zambella [28], found a new direction in this investigation. They introduced tracing as a key concept that clarifies and calibrates lowness notions.

Definition 1.1. A *trace* for a partial function $\psi\colon \omega \to \omega$ is a sequence $\langle S_n \rangle$ of finite sets such that for all $n \in \operatorname{dom}\psi$, $\psi(n) \in S_n$.

Thus, a trace for a partial function ψ indirectly specifies the values of ψ by providing finitely many possibilities for each value; it provides a way of "guessing" the values of the function ψ. Such a trace will be useful if it is easier to compute than the function ψ. In general, ψ will not be computable, but the trace $\langle S_x \rangle$ will be computable or uniformly c.e.:

Definition 1.2. Let $\langle S_n \rangle$ be a trace for a partial function ψ.

Both authors are supported by the Marsden Fund of New Zealand. Additionally the first author was supported by a James Cook Fellowship.

[1] Here K denotes prefix-free Kolmogorov complexity. A *set*, or a *real*, is an element of Cantor space 2^ω. If S is any set and $h, g\colon S \to \omega$, we say that $h \leqslant^+ g$ if there is some $d < \omega$ such that for all n, $h(n) \leqslant g(n) + d$. We assume that the reader is familiar with the rudiments of algorithmic randomness as described in initial segments of Downey and Hirschfeldt [6], Nies [23], or Li-Vitanyi [16]. We will be following the notation of Downey and Hirschfeldt [6].

(1) The trace is *computable* if the function $n \mapsto S_n$ is computable, that is, if there is a computable function g such that for all n, $S_n = D_{g(n)}$, where $\langle D_m \rangle$ is an effective enumeration of all finite sets of numbers.
(2) The trace is *c.e.* if the sequence $\langle S_n \rangle$ is uniformly c.e., that is, if there is a computable function g such that for all n, $S_n = W_{g(n)}$, where $\langle W_m \rangle$ is an effective enumeration of all computably enumerable sets of numbers.

We can, for example, recast classical concepts in the language of traces: a Turing degree $\mathbf{a}$ is hyperimmune-free if and only if every (total) function g computable from $\mathbf{a}$ has a computable trace. Terwijn and Zambella used a uniform version of hyperimmunity to characterise a notion of lowness.

Definition 1.3. An *order function* is a nondecreasing, computable and unbounded function h such that $h(0) > 0$. If h is an order function and $\langle S_n \rangle$ is a trace, then we say that $\langle S_n \rangle$ is an *h-trace* (or that $\langle S_n \rangle$ is bounded by h) if for all n, $|S_n| \leqslant h(n)$.

Terwijn and Zambella showed that a real A is low for Schnorr null tests if and only if there is some order function h such that every (total) function computable from A has a computable h-trace. This was later extended by Kjos-Hanssen, Nies and Stephan[14] to show that such reals are exactly those that are low for Schnorr randomness.

Zambella (see Terwijn [26]) observed that if A is low for Martin-Löf randomness then there is an order function h such that every function computable from A has a c.e. h-trace. This was improved by Nies [20], who showed that one can replace total by partial functions. In some sense it is natural to expect a connection between uniform traceability and lowness notions such as K-triviality; if every function computable (or partial computable) from A has a c.e. h-trace, for some slow-growing order function h, then the value of such a function on input n can be described by a little more than $\log n + \log h(n)$ many bits.

The question arises: is all of this merely an interesting footnote to the study of algorithmic randomness? Or might it be possible to understand lowness and more generally randomness using purely computability-theoretic tools such as tracing?

A test question is whether K-triviality could be characterised by traceability, in a similar way to the characterisation of lowness for Schnorr randomness which was discussed above. In particular, is there a set $\mathcal{F}$ of order functions such that a set A is K-trivial if and only if for all $h \in \mathcal{F}$, every A-computable function has a c.e. h-trace?

This problem remains open. However, an attempt toward a solution lead to the introduction of what seems now a fairly fundamental concept, which is not only interesting in its own right, but now has been shown to have deep connections with randomness.

Definition 1.4 (Figueira, Nies and Stephan [9])**.** Let h be an order function. A real A is *h-jump-traceable* if every A-partial computable function has a c.e. h-trace. A real is *strongly jump-traceable* if it is h-jump-traceable for every order function h.

Figueira, Nies and Stephan gave a "cost function" construction of a strongly jump-traceable c.e. set. Answering a question from Miller and Nies [17], together with Peter Cholak, in part I of this paper [2], the authors showed that the strongly jump-traceable c.e. sets form a proper subclass of the c.e. K-trivial reals. They also showed that the class forms an ideal in the c.e. degrees. This ideal was later shown to be Π^0_4 complete by Ng [18, 19], giving an alternative proof of the proper containment, as the K-trivial c.e. sets form a Σ^0_3 ideal. The paper [2] introduced new combinatorial tools for dealing with the class of c.e., strongly jump-traceable sets, collectively known as the "box-promotion" technique.

Subsequently, the class of c.e., strongly jump-traceable sets has been shown to have remarkable connections with randomness. Greenberg, Hirschfeldt, and Nies [10] proved that a c.e. set is strongly jump-traceable if and only if it is computable from every superlow random sets, if and only if it is computable from every superhigh random set. Kučera and Nies [15] showed that every c.e. set which is computable from a Demuth random set is strongly jump-traceable, relating such random sets with the "benign cost functions" which by work of Greenberg and Nies [11] characterise c.e., strong jump-traceability. Other attractive spin-offs in the arena of randomness include Nies's new work on the calculus of cost functions [22]. This material is only just beginning to work itself out and we expect a lot more to grow from these ideas.

Additionally, strongly jump-traceable sets have proven to have applications in classical computability. To illustrate, relativising the construction of Figueira, Nies and Stephan of a noncomputable, strongly jump-traceable c.e. set yields a pseudo-jump operator, which when inverted (Jockusch and Shore [12, 13]) yields an incomplete c.e. set A relative to which $\emptyset'$ is strongly jump-traceable. Such sets are very high, in that they resemble $\emptyset'$: they are all almost everywhere dominating in the sense of Dobrinen and Simpson[2] [4], and so are superhigh [24]. In [5], the authors have recently solved a question of Coles, Downey, Jockusch and LaForte [3] implicitly going back to Jockusch and Shore [12], by showing that the pseudo-jump operator which constructs a relative strongly jump-traceable set cannot be inverted with avoiding upper cones in the c.e. Turing degrees. Indeed, there is a noncomputable c.e. set which is computable from *all* c.e. sets relative to which $\emptyset'$ is strongly jump-traceable.

All of this brings us to the present paper. Notable is the lack in all of this material of anything about non-c.e. strongly jump-traceable reals. In fact, Nies has shown that there is some order function h such that there are continuum many h-jump-traceable reals. This result could discourage attempts to show that all strongly jump-traceable sets are K-trivial.

Up to now, no adaptation of the box-promotion technique for non-c.e. sets has been developed. This is a combinatorial technique that allows utilisation of the assumption that a c.e. set A is h-jump-traceable for some slow-growing order function h; the gist of it is that while trying to ascertain that some string is an initial segment of A, any mistake is beneficial as it allows us to limit the number of future mistakes on a large number of future tests. In the c.e. case, the constructions make heavy usage of the fact that a computation from a c.e. oracle, once destroyed, can never come back, clearing the deck for future computations; this is not so in the Δ^0_2 case. Translated to box-promotion, the problem is that in the c.e. case, when boxes are promoted, they remain promoted for ever, while in the Δ^0_2 case, boxes can be demoted, introducing seemingly unsurpassable problems. Moreover, in the general case we do not even assume that the given strongly jump-traceable set is

[2] A is almost everywhere dominating if for almost all reals X, every function computable from X is dominated by some function computable from A.

Δ^0_2, so no approximation to the set is given, so it is not at all clear what it is that we trace.

In this paper we show how to overcome these difficulties, and prove the following theorem.

Theorem 1.5. *There is a computable order function h such that every h-jump-traceable set is K-trivial.*

The order function h grows fairly slowly, at a rate similar to the double logarithm.

Beyond this result, much is unknown. Do the strongly jump-traceable sets form an ideal? Can they be characterised by randomness? The strongest conjecture is the following:

Conjecture 1.6. *Every strongly jump-traceable set is computable from some c.e., strongly jump-traceable set.*[3]

Despite removing the assumption of computable enumerability, the construction in the current paper is similar to the earlier box-promotion constructions in that it is "adaptive", that is, definition of the A-partial computable function depends on what values show up in the trace of the function. This severely restricts the applicability of this proof to an oracle version which is obtained by partial relativisation. Nies (see [23]) defined a weak reducibility associated with strong jump-traceability: for reals A and B, $A \leqslant_{\mathrm{SJT}} B$ if for every order function h, every A-partial computable function has a B-c.e. h-trace. This partial relativisation (A instead of $A \oplus B$, computable order functions instead of B-computable order functions) is necessary to make this relation transitive. Similarly, the weak reducibility associated with K-triviality (really, with lowness for randomness / for K) is LR-reducibility, where $A \leqslant_{\mathrm{LR}} B$ if every B-random set is A-random, or equivalently, if $K^B \leqslant^+ K^A$. The difficulty in partially relativising an adaptive construction means that we still do not know the answer to the following question, a strengthening of Theorem 1.5:

Question 1.7. *Does $A \leqslant_{\mathrm{SJT}} B$ imply $A \leqslant_{\mathrm{LR}} B$?*

2. Solovay functions and k-trees

The rest of the paper is devoted to the proof of Theorem 1.5.

Recall that a real A is K-trivial if $K(A\upharpoonright_n) \leqslant^+ K(n)$. To simplify our presentation, we replace the right hand side by a computable function. Recall that by the minimality of K, if g is any computable function, then $K \leqslant^+ g$ if and only if the sum $\sum_n 2^{-g(n)}$ is finite.

Definition 2.1. A *Solovay* function is a computable function g such that $K \leqslant^+ g$ and there is an infinite set S such that $g\upharpoonright_S \leqslant^+ K\upharpoonright_S$. [4]

Solovay [25] showed the existence of a Solovay function. More recently, Downey and Bienvenu gave a systematic study of these functions.

[3] This conjecture would imply that every strongly jump-traceable set obeys every benign cost function, which is also an open problem.

[4] Recall again that this means that for some constant d, for all $n < \omega$ we have $K(n) \leqslant g(n) + d$, but that $|K(n) - g(n)|$ is bounded on S.

Theorem 2.2.

(1) *Let g be a computable function such that $K \leqslant^+ g$. Then g is a Solovay function if and only if the number $\sum_n 2^{-g(n)}$ is Martin-Löf random.*
(2) *There is a Solovay function g such that for all reals A, A is K-trivial if and only if $K(A\restriction_n) \leqslant^+ g(n)$.*

We remark that in recent research, Bienvenu, Merkle and Nies extended part (2) of Theorem 2.2 and showed the equivalence holds for any Solovay function.

We fix, therefore, a Solovay function g, so Theorem 2.2(2) holds for g. By changing g by an additive constant, we assume that $\sum_n 2^{-g(n)} < 1$.

Definition 2.3. For $n < \omega$, let $c(n) = \sum_{m\leqslant n} 2^{-g(m)}$.

The idea is that if we believe that a string σ is an initial segment of A, then we believe that every initial segment of σ is also an initial segment of A, and so the "cost" of asking for short descriptions of σ and all of its initial segments is $c(|\sigma|)$. We let $c(\omega) = \sum_n 2^{-g(n)}$. This is a left-c.e. real: the collection of rational numbers $q < c(\omega)$ is computably enumerable.

The general plan is to enumerate a tree T which consists of strings which have short descriptions and which contains every initial segment of A. To ensure that indeed every string on T has a short description, we use a KC (bounded request) set; so we need to show that the total cost of all strings on T is finite. This will be obtained by limiting the size of finite subsets of T which are determined by granularity of cost: for each k, the restriction of T to strings of lengths whose cost passes an integer multiple of 2^{-k}.

Definition 2.4. Let $q < c(\omega)$ be a binary rational number. We let n_q be the least natural number n such that $c(n) \geqslant q$.

Thus the function $n \mapsto n_q$ is partial computable, as its domain is c.e. (and not computable). Note that $n_0 = 0$.

For $k < \omega$, let

$$Q_k = \left\{0, 2^{-k}, 2\cdot 2^{-k}, 3\cdot 2^{-k}, \dots, 2^{k-1}\cdot 2^{-k}, 1\right\};$$

so the set of binary rationals in the interval $[0,1]$ is $\bigcup_k Q_k$. We let

$$N_k = \{n_q \,:\, q \in Q_k, q < c(\omega)\}.$$

Definition 2.5. Let $k < \omega$. A k-tree is a set of strings T such that:

(1) For all $\sigma \in T$, $|\sigma| \in N_k$.
(2) If $\sigma \in T$, $n \in N_k$ and $n < |\sigma|$, then $\sigma\restriction_n \in T$.

Note that for all k, N_k is finite, and so every k-tree is finite.

Ordered by extension, a k-tree is a graph-theoretic tree, and so we call an element of such a tree T with no proper extension in T a *leaf* of T.

Proposition 2.6. *Let A be strongly jump-traceable. Then there is some $e < \omega$ and a uniformly c.e. sequence $\langle T_k\rangle_{k\geqslant e}$ such that:*

(1) *every T_k is a k-tree;*
(2) *for all $k > e$, if $\sigma \in T_k$ and $|\sigma| \in N_{k-1}$ then $\sigma \in T_{k-1}$;*
(3) *for all k, T_k has at most k many nonempty leaves; and*

(4) for all k, for all $n \in N_k$, $A\upharpoonright_n \in T_k$.

Proposition 2.6 details the combinatorial heart of the proof. It will be proved in the next section, using a "generalised" box-promotion argument. We now show that it is sufficient to prove the main theorem.

Proof of Theorem 1.5, given Proposition 2.6. Let $T = \bigcup_{k \geqslant e} T_k$. Our goal is to show that

$$\sum_{\sigma \in T} 2^{-g(|\sigma|)}$$

is finite, for then we could issue short descriptions for all strings on T, and so for all initial segments of A.

Lemma 2.7. *For all $n < \omega$, $A\upharpoonright_n \in T$.*

Proof. This follows from the fact that $\bigcup_{k \geqslant e} N_k = \omega$. Let $n > 0$. Since $c(n) > c(n-1)$, the interval $(c(n-1), c(n)]$ contains some binary rational number q. Then $n = n_q$. Since $q \leqslant c(n)$ we have $q < c(\omega)$. There is some $k \geqslant e$ such that $q \in Q_k$, so $n \in N_k$. □

Fix $k \geqslant e$. For every $\sigma \in T_k$, we let $q_k(\sigma)$ be the least $q \in Q_k$ such that $|\sigma| = n_q$. If σ is nonempty, then $q_k(\sigma) > 0$, and $q_k(\sigma)$ is the unique $q \in Q_k$ such that $|\sigma| = n_q$ and $n_q \neq n_{q-2^{-k}}$. For nonempty $\sigma \in T_k$, we let

$$S_k(\sigma) = \left\{ \tau \,:\, \sigma\upharpoonright_{n_{q_k(\sigma)-2^{-k}}} \subset \tau \subseteq \sigma \right\};$$

note that $\sigma\upharpoonright_{n_{q_k(\sigma)-2^{-k}}}$ is σ's immediate predecessor on T_k. For the empty string λ we let $S_k(\lambda) = \{\lambda\}$.

For $k > e$, let L_k be the collection of nonempty leaves σ of T_k such that $|\sigma| \in N_k \setminus N_{k-1}$; by property (2), L_k is the set of leaves σ of T_k such that $\sigma \notin T_{k-1}$. We let $L_e = T_e$.

Lemma 2.8. *Let $k > e$. For any $\sigma \in L_k$,*

$$\sum_{\tau \in S_k(\sigma)} 2^{-g(|\tau|)} < 2^{-k+1}.$$

Proof. Let $q = q_k(\sigma)$; since σ is nonempty, $q > 0$. The sum is

$$\sum_{\tau \in S_k(\sigma)} 2^{-g(|\tau|)} = \sum \left\{ 2^{-g(l)} \,:\, n_{q-2^{-k}} < l \leqslant n_q \right\} = c(n_q) - c(n_{q-2^{-k}}).$$

Now by definition, we have $c(n_{q-2^{-k}}) \geqslant q - 2^{-k}$. We also assumed that $n_q \notin N_{k-1}$. Hence $q \notin Q_{k-1}$. This implies that $q + 2^{-k} \in Q_k$; so either $q + 2^{-k} > q(\omega)$ or $n_{q+2^{-k}} > n_q$. In the former case, we have $c(n_q) < c(\omega) < q + 2^{-k}$; in the latter case, we have, by minimality of $n_{q+2^{-k}}$, $c(n_q) < q + 2^{-k}$. In either case we see that $c(n_q) < q + 2^{-k}$. Together with $c(n_{q-2^{-k}}) \geqslant q - 2^{-k}$ we get $c(n_q) - c(n_{q-2^{-k}}) < 2 \cdot 2^{-k}$ as required. □

For $k \geqslant e$, let

$$G_k = \bigcup_{\sigma \in L_k} S_k(\sigma).$$

Lemma 2.8 and property (3) imply that for $k > e$,

$$\sum_{\tau \in G_k} 2^{-g(|\tau|)} \leqslant k2^{-k+1}.$$

Let $G = \bigcup_{k \geqslant e} G_k$. Then

$$\sum_{\tau \in G} 2^{-g(|\tau|)} \leqslant \sum_{\tau \in G_e} 2^{-g(|\tau|)} + \sum_{k>e} 2k2^{-k}$$

which is finite because G_e is finite.

Lemma 2.9. *$T \subseteq G$.*

(In fact, $T = G$; the argument of Lemma 2.7 can be used to show that T is closed under taking initial segments.)

Proof. By induction on $k \geqslant e$, we show that for all $\sigma \in T_k$, $S_k(\sigma) \subset G$. The lemma follows, since for all σ and k, $\sigma \in S_k(\sigma)$.

For $k = e$, we defined $L_e = T_e$ so for all $\sigma \in T_e$, $S_e(\sigma) \subseteq G_e$.

Let $k > e$, and let $\sigma \in T_k$ be nonempty. Let $q = q_k(\sigma)$. There are three cases:

- If $n_q \in N_{k-1}$, then by property (2), $\sigma \in T_{k-1}$. Certainly
$$\sigma\restriction_{n_{q-2^{-(k-1)}}} \subseteq \sigma\restriction_{n_{q-2^{-k}}},$$
so $S_k(\sigma) \subseteq S_{k-1}(\sigma)$. By induction, $S_{k-1}(\sigma) \subset G$, so $S_k(\sigma) \subset G$.
- If $n_q \notin N_{k-1}$ and σ is a leaf of T_k, then $\sigma \in L_k$, in which case by definition of G_k, we have $S_k(\sigma) \subset G$.
- If $n_q \notin N_{k-1}$ and σ is not a leaf of T_k, then there is some $\rho \in T_k$ extending σ such that $|\rho| = q + 2^{-k}$. Since $n_q \notin N_{k-1}$ we have $q \notin Q_{k-1}$, so $q + 2^{-k} \in Q_{k-1}$; since $n_{q+2^{-k}}$ is defined, we have $n_{q+2^{-k}} \in N_{k-1}$. By property (2), $\rho \in T_{k-1}$. We have $\rho\restriction_{n_{q+2^{-k}-2^{-(k-1)}}} = \rho\restriction_{n_{q-2^{-k}}}$, so $S_k(\sigma) \subset S_{k-1}(\rho)$, so by induction, $S_k(\sigma) \subset G$. □

It follows that indeed

$$\sum_{\sigma \in T} 2^{-g(|\sigma|)}$$

is finite. Since T is c.e., by the KC Theorem, there is a constant d such that for all $\sigma \in T$, $K(\sigma) \leqslant g(|\sigma|) + d$. By Lemma 2.7, for all n, $K(A\restriction_n) \leqslant g(n) + d$. The choice of g shows that A is K-trivial. This completes the proof of Theorem 1.5. □

3. A non-c.e. box-promotion argument

It remains to prove Proposition 2.6. As promised, this is done by a box-promotion argument. Let A be strongly jump-traceable.

3.1. **The overhead of the Recursion Theorem.**

As standard in a box-promotion argument, we define a partial computable functional Ψ, and by the Recursion (fixed point) Theorem, we obtain a c.e. trace $\langle S_z \rangle$ for Ψ^A. The trace will be bounded by a computable function f that we define ahead of time, except for finitely many inputs due to "overhead" from the Recursion Theorem: the Recursion Theorem tells us that essentially, we control a column of the jump of A, but we do not choose the index of this column; because of this, we cannot expect to bound the size of $\langle S_z \rangle$ as we wish for all inputs z, but only for almost all of them.

Formally, this is done as follows. Let $\langle\Psi_e\rangle$ be an enumeration of all partial computable functionals; recall that J is a partial computable functional such that for all $X \in 2^\omega$, J^X is a universal A-partial computable function.

Lemma 3.1 (Lemma 1.5 of Cholak, Downey, and Greenberg [2])**.** *Let f be an order function. There is an order function h such that for all A, from a (c.e. index for a) c.e. h-trace for J^A we can obtain, uniformly in e, a (c.e. index for a) c.e. $\max\{e, f\}$-trace for Ψ_e^A.*

By the Recursion Theorem we obtain an index e such that $\Psi = \Psi_e$; this e is fixed for the rest of the construction. We also fix, ahead of time, a c.e. h-trace for J^A. Lemma 3.1 then yields a c.e. trace $\langle S_z\rangle$ for Ψ^A, which is bounded by $\max\{e, f(n)\}$.

Also ahead of time, we define a partition $\langle I_k\rangle$ of ω into intervals, which is of course determined by specifying $|I_k|$ for each k. This defines f since we let, for all k, for all $z \in I_k$, $f(z) = k$. Thus, for all $k \geqslant e$, for all $z \in I_k$, we have $|S_z| \leqslant k$.

3.2. **The general idea.** As in other constructions, we use Ψ and the trace $\langle S_z\rangle$ to *test* possible initial segments of A. Recall that the purpose is to enumerate the sequence of k-trees $\langle T_k\rangle$ of Proposition 2.6. If A were computable, we could have of course let T_k be the single branch consisting of initial segments of A. Since A is not computable, we need to enumerate several strings into T_k. With an associated degree of confidence, we believe they may be initial segments of A. We need to ensure that the correct initial segments are enumerated, but that not too many strings are enumerated.

To *test* a string σ on input z, we let $\Psi^\sigma(z) = \sigma$. The test is successful if $\sigma \in S_z$. If σ is an initial segment of A, then the test will be successful. Note that consistency of Ψ means that we cannot test comparable strings on the same input.

Unlike the c.e. case, we do not have an approximation for A, and so we need to test arbitrary strings. The combinatorial heart of the argument is the mechanism for deciding which strings are tested on which inputs. The general aim is to "make the opponent pay dearly" for successful tests of wrong initial segments. To ensure that each T_k has at most k many leaves, we will guess the sets of leaves of each level of T_k, and in that way amplify (or promote) boxes.

3.3. **Simplified versions.** To give a better idea of how the construction works, we first describe unrealistically easy cases.

One level. Let $k = 0$, so $Q_k = \{0, 1\}$; and for convenience, make the false assumption that $c(\omega) > 1$. Let $n = n_1$; so T_0 should contain the empty string, and strings of length n. Of course it is impossible to have both $A\restriction_n \in T_0$ and have T_0 have at most 0 leaves. But we show how to get $A\restriction_n \in T_0$ and make the number of leaves of T_0 bounded by e, where again e, defined above, is the "overhead" charged by the Recursion Theorem. Without knowing A at all, we simply test all strings of length n on an input $z \in I_e$, so $|S_z| \leqslant e$. We then enumerate all the strings of length n that show up in S_z to T_0. Since $\langle S_z\rangle$ traces Ψ^A, and $\Psi^A(z) = A\restriction_n$, we get the desired tree.

Two levels. Now let $k = 1$, so $Q_k = \{0, 1/2, 1\}$; and again make the false assumption that $c(\omega) > 1$. For abbreviation, let $n = n_{1/2}$ and $m = n_1$; we assume that $n < m$. Our aim is to define T_1 so that $A\restriction_n, A\restriction_m \in T_1$, but that T_1 has at most e many leaves.

We can easily ensure that T_1 contains at most e many strings of length n, and at e many strings of length m; we adopt the technique of the case $k = 0$, this time using two inputs $z, w \in I_e$, test all strings of length n at z and all strings of length m at w. Indeed we do so, and so get two lists of strings, say $\sigma_1, \sigma_2, \dots$ and $\tau_1, \tau_2, \dots$, of strings of lengths n and m respectively; both lists comprise of at most e strings, and $A\restriction_n$ appears on the first list, $A\restriction_m$ on the second list. From these two lists we need to winnow sublists, which would make up T_1, so that the initial segments of A are not omitted, but overall we get at most e many leaves altogether.

First of all, we need to ensure that T_1 is a tree. For this, we may assume that for every τ_i on the second list, $\tau_i\restriction_n$ appears on the first list as some σ_j. This is easily done by holding off enumerating τ_i until we see $\tau_i\restriction_n$ enumerated as some σ_j. Below, we will say that τ_i has been *preapproved* (Definition 3.2) to appear on the list of τ's. Hence, for all i such that τ_i is defined, there is some $j(i)$ such that $\sigma_{j(i)}$ is defined and $\sigma_{j(i)} \subset \tau_i$. Since the collection of all strings of length n forms an antichain, for all $j \neq j(i)$, σ_j is incomparable with τ_i; the importance of this is that for $j \neq j(i)$, τ_i and σ_j can be both tested on the same input, whereas τ_i and $\sigma_{j(i)}$ cannot.

The idea now is to amplify the evidence that each σ_j is potentially an initial segment of A. We assign a large set $M \subset I_e$ of inputs z for the task of defining T_1 (below it is denoted by $M^1(\lambda)$). For each j, we test σ_j on exactly half of the inputs in M. These are distributed independently, so for any finite subset B of $\{\sigma_1, \sigma_2, \dots\}$, the collection of all inputs $z \in M$ on which exactly the elements of B are tested has size $|M|/2^e$. Each such set B represents a guess as to what the leaves of T_1 of length n are going to be. We enumerate σ_j into T_1 if all tests of σ_j are successful, that is, if for all $z \in M$ such that σ_j is tested on z, we see that $\sigma_j \in S_z$.

From the point of view of a string τ_i, any set B as above that contains $\sigma_{j(i)}$ cannot be a correct guess for a set of leaves of T_1. However, any such set that omits $\sigma_{j(i)}$ is possibly correct. Hence we can be maximal: we test τ_i on every $z \in M$ on which $\sigma_{j(i)}$ is not tested. This ensures that Ψ is consistent. We enumerate successful τ_i's into T_2. Now let B be the collection of leaves of T_1 of length n, and C be the collection of leaves of T_1 of length m. The distribution of testing inputs in M ensures that there is some $z \in M$ on which all the strings of B and C are tested. Since $z \in I_e$, this ensures that $|B \cup C| \leqslant e$ as required.

The actual construction is a generalisation, to more levels of the case of two levels; the main ideas are the same. These ideas are also sufficient to show that the trees T_k cohere with each other (Property (2) of Proposition 2.6).

Where is the promotion? While this construction is not as dynamic as a c.e.-box-promotion argument, we can think of it as follows. Again let B be the collection of leaves of T_1 of length n. The elements of T_1 are not initial segments of A, since they have no extension on T_1 of length m. Let $M^1(B)$ be the collection of inputs $z \in M$ at which exactly strings in B (among all the σ_j's) are tested. The fact that each $\sigma_j \in B$ is enumerated into T_1 means that these tests were successful: our opponent spent some of their capital on giving us wrong information. The opponent has wasted $|B|$ many possibilities of enumerating strings into S_z for $z \in M^1(B)$. We say that the "box" $M^1(B)$ has been *promoted* from level e to level $e - |B|$. We can now test strings of length m from the τ_i's, including the correct initial segment of

A, on $z \in M^1(B)$, and the opponent can only deceive us with $e - |B| - 1$ many incorrect such strings.

3.4. **The construction.** We let $|I_k| = 2^k + 1 + (2^k)^{2^k}$. For $q \in Q_k$, fix (distinct) $z(k,q) \in I_k$. We will test all strings of length n_q on $z(k,q)$, to make the first step of whittling down the collection of strings of length n_q that may be enumerated to T_k, from all such strings to just k many.

We let
$$M^k(\lambda) = I_k \setminus \{z(k,q) \,:\, q \in Q_k\};$$
so $|M^k(\lambda)| = (2^k)^{2^k}$. Now we define $M^k(\nu)$ for every ν which is a sequence of subsets of $\{1,2,\dots,k\}$ of length at most 2^k. $M^k(\lambda)$ has just been defined; if ν is a sequence of subsets of $\{1,2,\dots,k\}$ and $|\nu| < 2^k$, and $M^k(\nu)$ is defined, then by induction,
$$|M^k(\nu)| = (2^k)^{2^k - |\nu|},$$
and we let
$$\left\{M^k(\nu B) \,:\, B \subseteq \{1,2,\dots,k\}\right\}$$
be a partition of $M^k(\nu)$ into 2^k subsets of equal size, namely $(2^k)^{2^k-|\nu|-1} = (2^k)^{2^k-|\nu B|}$. Note that if $\nu \subset \mu$ then $M^k(\mu) \subset M^k(\nu)$, but if ν and μ are not comparable, then $M^k(\nu)$ and $M^k(\mu)$ are disjoint.

Let
$$\bar{Q}_k = \left\{q \in Q_k \,:\, q > 0,\ q < c(\omega)\ \&\ n_q > n_{q-2^{-k}}\right\}.$$
$\bar{Q}_k$ is c.e., uniformly in k.

As discussed above, for a set B of indices of strings of length n_{q_1} where $q_1 = \min \bar{Q}_k$, $M^k(B)$ will be a testing ground for the strings that believe that B can be a set of indices of leaves of T_k of length n_{q_1}; for another set C of indices of strings of length some n_{q_2}, where q_2 is the next element of $\bar{Q}_k$, $M^1(B,C)$ will be a testing ground for strings that believe that B is a set of indices for leaves of T_k of length n_{q_1} *and* C is a set of indices for leaves of T_k of length n_{q_2}, and so on.

Initial testing. For all $k \geqslant e$, for all $q \in \bar{Q}_k$, we test all binary strings of length n_q at input $z(k,q)$. Note that indeed all of these strings are incomparable, so this test keeps Ψ consistent.

Approval and general testing. By induction on $k \geqslant e$, and then by induction on $q \in \bar{Q}_k$, we describe how to further test strings that show up in $S_{z(k,q)}$; and decide which strings of length n_q are enumerated into T_k (we also enumerate the empty string into every T_k).

For $k \geqslant e$ and $q \in \bar{Q}_k$, let $P(k,q)$ be the collection of all pairs (k',q') such that:

- $e \leqslant k' < k$, $q' \in \bar{Q}_{k'}$, and $n_{q'} \leqslant n_q$; or
- $k' = k$, $q' \in \bar{Q}_k$, and $q' < q$.

Note that for all $(k',q') \in P(k,q)$, $n_{q'} \leqslant n_q$. Since $\bar{Q}_k$ is not computable, the domain of the function $(k,q) \mapsto P(k,q)$ is not computable either. Nonetheless, this function is partial computable. Once we know that $q \in \bar{Q}_k$, we can effectively compute all of $P(k,q)$. The point is that if we know that $q \in Q_k$ and $q < c(\omega)$, then since we can calculate n_q and so $c(n_q)$, for all $k' < k$ we know the set of $q' \in Q_{k'}$ such that $n_{q'} \leqslant n_q$, and for all such q', we know that $q' < c(\omega)$.

The point is that $P(k,q)$ is the collection of pairs (k',q') such that every string $\sigma \in T_k$ of length n_q must have a (not necessarily proper) initial segment on $T_{k'}$ of length $n_{q'}$.

Definition 3.2. We say that a string $\sigma \in S_{z(k,q)}$ is (k,q)-*preapproved* if for all $(k',q') \in P(k,q)$, we have $\sigma\restriction_{n_{q'}} \in T_{k'}$.

The idea of this definition is to achieve the coherence between the trees T_k (Property (2) of Proposition 2.6) and to ensure that every T_k is in fact a tree. Until a string of length n_q is (k,q)-preapproved, it is not even a candidate for being enumerated into T_k.

We will see that for all $k \geqslant e$ and $q \in \bar{Q}_k$, the collection of (k,q)-preapproved strings is c.e., uniformly in k and q. To see this, we will require that the trees $\langle T_k \rangle_{k \geqslant e}$ are c.e., uniformly in k. To avoid circularity, we need to be precise. For $k \geqslant e$ and $q \in \bar{Q}_k$, the inductive hypothesis is that for all $(k',q') \in P(k,q)$, the collection of strings of length $n_{q'}$ on $T_{k'}$ is c.e., uniformly in k' and q'.[5] From this hypothesis we can conclude that the collection of (k,q)-preapproved strings is also c.e., uniformly in k and q. Our instructions below will then show that given all of this information, the collection of strings of length n_q on T_k is also c.e., uniformly, thus enabling the induction to continue.

Let $\sigma_1(k,q), \sigma_2(k,q), \ldots$ be an effective list of the (k,q)-preapproved strings. The list has length at most k, since $|S_{z(k,q)}| \leqslant k$. Note, though, that this is a c.e. list, in the sense that if fewer than k many such strings have been listed, we can never be sure that no more strings will be listed in the future.

Suppose that $\sigma = \sigma_i(k,q)$ is (k,q)-preapproved. Let

$$m(k,q) = \left|\left\{q' \in \bar{Q}_k \,:\, q' < q\right\}\right|;$$

note that $m(k,q) \leqslant 2^k$ and $m(k,q) \geqslant 1$. We further test σ on all the elements of several boxes $M^k(\nu)$, where ν is a sequence of subsets of $\{1,2,\ldots,k\}$ of length $m(k,q)$. Which such sequences? We test σ on the elements of $M^k(B_1,\ldots,B_{m(k,q)})$ if:

(1) $i \in B_{m(k,q)}$;
(2) For all $q' \in \bar{Q}_k$ such that $q' < q$, if $\sigma_j(k,q')$ is an initial segment of σ, then $j \notin B_{m(k,q')}$.

For the second condition, note that since σ is (k,q)-preapproved and $(k,q') \in P(k,q)$, and since $S_{z(k,q')}$ is an antichain of strings, there is a unique j such that $\sigma_j(k,q')$ is defined and is an initial segment of σ, and such a j is already present when we test σ.

Finally, if σ is a (k,q)-preapproved string, then we enumerate σ into T_k if for all z on which we tested σ we have $s \in S_z$, that is, all tests of σ are successful. Since $\langle S_z \rangle$ are uniformly c.e., and since the collection of (k,q)-preapproved strings is c.e., uniformly in k and q, we indeed get that the collection of strings of length n_q on T_k is c.e., uniformly, as required.

Further, we notice that the testing procedure keeps Ψ consistent:

Lemma 3.3. *No comparable strings are tested on an input z.*

[5] Or what amounts to almost the same, that for $k' \in [e,k)$, the trees $T_{k'}$ are c.e., uniformly in k'; and that for $q' \in \bar{Q}_k$ such that $q' < q$, the strings on T_k of length $n_{q'}$ are also c.e., uniformly in q'.

Proof. Let $z \in I_k$, and suppose that strings σ_0 and σ_1 are tested on z; we show that σ_0 and σ_1 are incomparable. We must have $z \in M^k(\lambda)$. There are $q_0, q_1 \in \bar{Q}_k$ and indices $i_0, i_1 \leqslant k$ such that $\sigma_0 = \sigma_{i_0}(k, q_0)$ and $\sigma_1 = \sigma_{i_1}(k, q_1)$; without loss of generality, $q_0 \leqslant q_1$. There is some $\nu = (B_1, \dots, B_m)$ of length $m = m(k, q_1)$ such that $z \in M^k(\nu)$.

Now there are two cases. If $q_0 = q_1$, then σ_0, σ_1 are both of length n_{q_0} and so are incomparable. Otherwise, $q_0 < q_1$; let $m_0 = m(k, q_0)$ which is smaller than m. There is a unique j such that $\sigma_j(k, q_0)$ is an initial segment of σ_1, and $j \notin B_{m_0}$. Now $\nu_0 = (B_1, \dots, B_{m_0})$ is the only sequence μ of subsets of $\{1, 2, \dots, k\}$ of length m_0 such that $z \in M^k(\mu)$; the fact that σ_0 is tested on z means that σ_0 is tested on $M^k(\nu_0)$, which in turn implies that $i_0 \in B_{m_0}$. Since $j \notin B_{m_0}$ we have $i_0 \neq j$, and so $\sigma_0 = \sigma_{i_0}(k, q_0)$ is not an initial segment of σ_1. Since $|\sigma_0| = n_{q_0} < n_{q_1}$, we conclude that σ_0 and σ_1 are incomparable. ☐

3.5. **Verification.** We show that the sequence $\langle T_k \rangle_{k \geqslant e}$ satisfies the properties required by Proposition 2.6.

By construction, the sequence $\langle T_k \rangle_{k \geqslant e}$ is uniformly c.e., and each T_k consists of strings of length n_q for some $q \in Q_k$. We first establish properties (1) and (2).

Lemma 3.4. *Let $\sigma \in T_k$.*

(1) If $n \in N_k$ and $n < |\sigma|$, then $\sigma \restriction_n \in T_k$.
(2) If $k > e$ and $|\sigma| \in N_{k-1}$, then $\sigma \in T_{k-1}$.

Proof. We have $N_k = \{0\} \cup \{n_q \; q \in \bar{Q}_k\}$. Let $q \in \bar{Q}_k$ such that $|\sigma| = n_q$. Now σ is (k, q)-preapproved: for all $(k', q') \in P(k, q)$ we have $\sigma \restriction_{n_{q'}} \in T_{k'}$.

If $n \in N_k$, $n < |\sigma|$, then either $n = 0$, in which case certainly $\sigma \restriction_n \in T_k$; or $n = n_{q'}$ for some $q' \in \bar{Q}_k$ smaller than q. In this case we have $(k, q') \in P(k, q)$, and so $\sigma \restriction_n \in T_k$.

Now suppose that $k > e$ and $|\sigma| = n_q \in N_{k-1}$. Let $q' \in \bar{Q}_{k-1}$ such that $n_q = n_{q'}$; then $(k-1, q') \in P(k, q)$. It follows that $\sigma = \sigma \restriction_{n_{q'}} \in T_{k-1}$. ☐

We verify property (3):

Lemma 3.5. *For all $k \geqslant e$, T_k has at most k many leaves.*

Proof. Fix $k \geqslant e$. We show, in fact, that every antichain on T_k has size at most k; of course, the set of leaves of T_k is the largest antichain on T_k. So let L be an antichain on T_k.

Let $q^* = \max\{\bar{Q}_k\}$ and $m^* = m(k, q^*)$. For $m \leqslant m^*$, let

$$B_m = \{i \in \{1, 2, \dots, k\} \, : \, \sigma_i(k, q) \in L\},$$

where q is the unique element of $\bar{Q}_k$ such that $m = m(k, q)$.

Consider the sequence $\nu = (B_1, B_2, \dots, B_{m^*})$. Since L is an antichain, for all $q \in \bar{Q}_k$, every $\sigma \in L$ of length n_q is tested on $M^k(B_1, \dots, B_{m(k,q)})$, and so on every input in $M^k(\nu)$; and since such σ is actually enumerated into T_k, these tests must be successful. We conclude that for all $z \in M^k(\nu)$ we have $L \subseteq S_z$. The conclusion follows from the fact that $|S_z| \leqslant k$ for all $z \in I_k$. ☐

And finally, property (4):

Lemma 3.6. *For all $k \geqslant e$, for all $n \in N_k$, $A \restriction_n \in T_k$.*

Proof. We enumerated the empty string into every T_k, so it remains to show the lemma for every nonempty initial segment of A. For $k \geqslant e$ and nonzero $n \in N_k$, there is some $q \in \bar{Q}_k$ such that $n = n_q$; so we show that for all $k \geqslant e$, for all $q \in \bar{Q}_k$, $A\restriction_{n_q} \in T_k$. This is proved by induction on k, and then on q.

Let $k \geqslant e$ and $q \in \bar{Q}_k$. By induction, for all $(k', q') \in P(k, q)$, we have $A\restriction_{n_{q'}} \in T_{k'}$. Let $\sigma = A\restriction_{n_q}$. The string σ is tested on $z(k, q)$, and since $\sigma \subset A$ we have $\sigma \in S_{z(k,q)}$. Hence σ is (k, q)-preapproved. Now σ is tested on the elements of various $M^k(\nu)$'s, and since it is an initial segment of A, all these tests are successful, so σ is enumerated into T_k as required. □

References

[1] Laurent Bienvenu and Rod Downey. Kolmogorov complexity and Solovay functions. In S. Albers and J.-Y. Marion, editors, *26th Annual Symposium on Theoretical Aspects of Computer Science, (STACS 2009)*, volume 3 of *Leibniz International Proceedings in Informatics*, pages 147–158. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, Germany, Dagstuhl, Germany, 2009.

[2] Peter Cholak, Rod Downey, and Noam Greenberg. Strong jump-traceabilty I: The computably enumerable case. *Adv. Math.*, 217(5):2045–2074, 2008.

[3] R. Coles, R. Downey, C. Jockusch, and G. LaForte. Completing pseudojump operators. *Ann. Pure Appl. Logic*, 136(3):297–333, 2005.

[4] Natasha L. Dobrinen and Stephen G. Simpson. Almost everywhere domination. *J. Symbolic Logic*, 69(3):914–922, 2004.

[5] Rod Downey and Noam Greenberg. Failure of cone avoidance for the strong jump-traceability pseudojump operator. In preparation.

[6] Rod Downey and Denis Hirschfeldt, *Algorithmic Randomness and Complexity*, Springer-Verlag, 2010.

[7] Rod Downey, Denis R. Hirschfeldt, André Nies, and Sebastiaan A. Terwijn. Calibrating randomness. *Bull. Symbolic Logic*, 12(3):411–491, 2006.

[8] Rod Downey, Denis R. Hirschfeldt, André Nies, and Frank Stephan. Trivial reals. In *Proceedings of the 7th and 8th Asian Logic Conferences*, pages 103–131, Singapore, 2003. Singapore Univ. Press.

[9] Santiago Figueira, André Nies, and Frank Stephan. Lowness properties and approximations of the jump. *Ann. Pure Appl. Logic*, 152(1-3):51–66, 2008.

[10] Noam Greenberg, Denis Hirschfeldt, and André Nies. Characterizing the strongly jump traceable sets via randomness. Submitted.

[11] Noam Greenberg and André Nies. Benign cost functions and lowness properties. To appear in *Jour. of Symb. Logic.*

[12] Carl G. Jockusch, Jr. and Richard A. Shore. Pseudojump operators. I. The r.e. case. *Trans. Amer. Math. Soc.*, 275(2):599–609, 1983.

[13] Carl G. Jockusch, Jr. and Richard A. Shore. Pseudojump operators. II. Transfinite iterations, hierarchies and minimal covers. *J. Symbolic Logic*, 49(4):1205–1236, 1984.

[14] Bjørn Kjos-Hanssen, André Nies, and Frank Stephan. Lowness for the class of Schnorr random reals. *SIAM J. Comput.*, 35(3):647–657 (electronic), 2005.

[15] Antonin Kučera and André Nies. Demuth randomness and computational complexity. To appear in *Ann. Pure and Applied Logic.*

[16] Ming Li and Paul Vitányi. *An introduction to Kolmogorov complexity and its applications.* Texts and Monographs in Computer Science. Springer-Verlag, New York, 1993.

[17] Joseph S. Miller and André Nies. Randomness and computability: open questions. *Bull. Symbolic Logic*, 12(3):390–410, 2006.

[18] Keng Meng Ng. On strongly jump traceable reals. *Ann. Pure Appl. Logic*, 154(1):51–69, 2008.

[19] Keng Meng Ng. *Computability, Traceability and Beyond.* PhD thesis, Victoria University of Wellington, 2009.

[20] André Nies. Lowness properties and randomness. *Adv. Math.*, 197(1):274–305, 2005.

[21] André Nies. Eliminating concepts. In *Computational prospects of infinity. Part II. Presented talks*, volume 15 of *Lect. Notes Ser. Inst. Math. Sci. Natl. Univ. Singap.*, pages 225–247. World Sci. Publ., Hackensack, NJ, 2008.
[22] André Nies. Calculus of cost functions. In preparation.
[23] André Nies. *Computability and Randomness.* Oxford Logic Guides. Oxford University Press, 2009.
[24] Stephen G. Simpson. Almost everywhere domination and superhighness. *MLQ Math. Log. Q.*, 53(4-5):462–482, 2007.
[25] R. Solovay. Draft of paper (or series of papers) related to Chaitin's work. IBM Thomas J. Watson Research Center, Yorktown Heights, NY, 215 pages, 1975.
[26] Sebastiaan A. Terwijn. *Computability and Measure.* PhD thesis, The Institute for Logic, Language and Computation (ILLC), University of Amsterdam, 1998.
[27] Sebastiaan A. Terwijn. Complexity and randomness. *Rendiconti del Seminario Matematico di Torino*, 62(1):1–38, 2004. Notes for a course given at the University of Auckland, March 2003.
[28] Sebastiaan A. Terwijn and Domenico Zambella. Computational randomness and lowness. *J. Symbolic Logic*, 66(3):1199–1205, 2001.

School of Mathematics, Statistics and Operations Research, Victoria University, P.O. Box 600, Wellington, New Zealand

E-mail address: Rod.Downey@vuw.ac.nz, Noam.Greenberg@vuw.ac.nz