A K-TRIVIAL SET WHICH IS NOT JUMP TRACEABLE AT CERTAIN ORDERS

DANIEL TURETSKY

Abstract. We construct a K-trivial c.e. set which is not jump traceable at any order in $o(\log x)$.

1. Introduction

The study of algorithmic randomness, specifically Martin-Löf randomness, has precipitated the study of those sets which are low for Martin-Löf randomness. This class was shown by Nies and his co-authors [7] to coincide with a number of other classes of computably weak sets, including the low for K sets and the K-trivial sets (for brevity, we will use the name K-trivial). For background on these notions, see the texts by Nies [8] or Downey and Hirschfeldt [3].

Motivated by the robust nature of this class, lowness for other randomness notions has been studied. Terwijn and Zambella [9] and then Kjos-Hanssen, Nies and Stephan [5] characterized those sets which are low for Schnorr randomness, while Bienvenu, Downey, Greenberg, Nies and Turetsky [1] characterized those sets which are low for Demuth randomness. In both cases, a notion of traceability was used to give a purely combinatorial characterization of the class. This stands in contrast to the K-trivial sets; although many different characterizations are known, they all involve effective randomness or measure in some fashion.

A (partial) function $f : \omega \to \omega$ is *traced* by a sequence of sets $\langle T(x)\rangle_{x\in\omega}$ if for every $x \in \operatorname{dom} f$, $f(x) \in T(x)$. Various traceability notions are arrived at by placing size restrictions or effectiveness restrictions on the sequence $\langle T(x)\rangle_{x\in\omega}$. For example:

Definition 1. A set A is *computably traceable* if every total function $f \leqslant_T A$ is traced by a sequence $\langle T(x)\rangle_{x\in\omega}$ with $|T(x)| = x + 1$ and canonical indices for the $T(x)$ can be given computably in x.

The computably traceable sets turn out to be precisely the sets which are low for Schnorr randomness (this is the above mentioned characterization). The characterization of low for Demuth randomness is similar, although slightly more complicated to state.

Towards a combinatorial characterization of K-triviality, jump traceability has been put forth as a candidate.

Definition 2. A *computable order* is a total, computable, non-decreasing, unbounded function.

The author was supported as a postdoctoral fellow by a Marsden grant from the Royal Society of New Zealand.

A set A is *jump traceable at order h* if every partial A-computable function f^A is traced by a sequence $\langle T(x)\rangle_{x\in\omega}$ with $|T(x)| \leqslant h(x)$ and c.e. indices for the $T(x)$ can be given computably in x

This is actually many traceability notions, depending on the choice of order h. Clearly slower growing orders result in a more restrictive notion. So we are left with studying the relationship with K-triviality for different orders h. Cholak, Downey and Greenberg [2] showed that every c.e. set which is jump traceable at order $\sqrt{\log x}/9$ is K-trivial. Meanwhile, Hölzl, Kräling and Merkle [4] showed a result which has the following as a corollary:

Proposition 3. *For every K-trivial set A, there is a constant d such that A is jump traceable at order $d\log x$.*

In this paper, we show that this result is, basically, tight. That is, we construct a K-trivial c.e. set which is not jump traceable at any computable order $h \in o(\log x)$. So while the question of a jump traceability characterization for K-triviality remains open, we know that, up to a multiplicative constant, the only possible candidate is $\log x$.

2. Preliminaries

The important property of orders in $o(\log x)$, which this proof depends upon, is the following:

Lemma 4. *If h is an order with $\sum_{x=0}^{\infty} 2^{-d\cdot h(x)} = \infty$ for all $d > 0$, then for any $c, N > 0$, there is an $n > N$ with $h(N + 2^{c\cdot n}) \leqslant n$.*

Note that every order in $o(\log x)$ satisfies the hypothesis; however, these are not the only orders which do. For example, define $a_1 = 2$ and $a_{n+1} = a_n + a_n^n$, and define $h(x) = \log a_n$ for all $x \in [a_n, a_{n+1})$. Then $h \notin o(\log x)$, but h still satisfies the hypothesis.

Proof. Fix c and N, and suppose $h(2^{(c+1)\cdot n}) > n$ for all but finitely many n. Then $h(x) > \frac{1}{c+1}\log x$ for all but finitely many x, and thus $\sum_{x=0}^{\infty} 2^{-2(c+1)\cdot h(x)} < \infty$, contrary to hypothesis. So $h(2^{(c+1)\cdot n}) \leqslant n$ for arbitrarily large n. But for sufficiently large n, $n > N$ and $2^{(c+1)\cdot n} > N + 2^{c\cdot n}$. □

Before we move on to the proof of the main result, we discuss the following function:

$$c(z, s) = \Omega_s - \Omega_z.$$

Here Ω is Chaitin's halting probability, and $\langle\Omega_s\rangle_{s\in\omega}$ is some computable increasing sequence of nonnegative rational numbers which converges to Ω. The reader may recognize this as an example of a cost function; cost functions are well studied, particularly in their relation to K-triviality (for a detailed treatment, see Nies's upcoming manuscript [6]). We do not discuss them in generality here, but instead only restate the result we need.

Definition 5. For $\langle A_s\rangle_{s\in\omega}$ an approximation to a Δ^0_2 set A, we define $c(A_s)$ to be $c(z_s, s)$, where z_s is least with $A_s(z_s) \neq A_{s+1}(z_s)$. If $A_s = A_{s+1}$, we take $c(A_s) = 0$.

We say that a set A *obeys c* if it has some computable approximation $\langle A_s\rangle_{s\in\omega}$ with $\sum_s c(A_s) < \infty$.

The important result is the following:

Lemma 6 (Nies [6]). *For a Δ^0_2 set A, A obeys c if and only if A is K-trivial.*

We will need one other fact:

Observation 7. *For a sequence $z_0 < s_0 < z_1 < s_1 < \cdots < z_n < s_n$,*

$$\sum_{i=0}^{n} c(z_i, s_i) < 1.$$

This follows by rearranging the terms in the sum:

$$\sum_{i=0}^{n} c(z_i, s_i) = \Omega_{s_n} - \Omega_{z_0} + \sum_{i=0}^{n-1} \left(\Omega_{s_i} - \Omega_{z_{i+1}}\right) \leqslant \Omega_{s_n} < \Omega < 1.$$

3. Proof of theorem

Now we are ready to prove the theorem.

Theorem 8. *There is a c.e. K-trivial set A which is not jump traceable at any computable order h with $\sum_{x=0}^{\infty} 2^{-d\cdot h(x)} = \infty$ for all $d > 0$. In particular, it is not jump traceable at any computable order $h \in o(\log x)$.*

Proof. We construct A in stages, ensuring that the approximation we construct obeys c, which will be sufficient to guarantee that A is K-trivial.

Let h^e be an enumeration of all (partial) computable orders, and let $\langle T^e_k(x)\rangle_{k\in\omega}$ be an enumeration of all h^e bounded c.e. traces. We construct a partial A-computable function f^A, and for every $k, e \in \omega$ we meet the following requirements:

P^e_k **::** If h^e is total with $\sum_{x=0}^{\infty} 2^{-d\cdot h^e(x)} = \infty$ for all $d > 0$, then there is an $x \in \omega$ with $f^A(x)\!\downarrow$ and $f^A(x) \notin T^e_k(x)$.

We partition the domain of f^A into infinitely many sets B^e, and work to meet all P^e_k-requirements on B^e. However, our choice of partition matters: for reasons that will become apparent later, each B^e must be an arithmetic progression. So we let $B^e = \{n \cdot 2^{e+1} + 2^e \mid n \in \omega\}$.

Basic strategy for P^e_k. The basic strategy for meeting requirement P^e_k is straightforward:

(1) Choose an $x \in B^e$.
(2) Wait until $h^e(x)\!\downarrow$.
(3) Define $f^A(x)$ to a large value with large use.
(4) Wait until $f^A(x) \in T^e_k(x)$.
(5) Enumerate an element into A below the use and redefine $f^A(x)$ to a large value with large use.
(6) Return to Step 4.

Since $|T^e_k(x)| \leqslant h^e(x)$, the above strategy can only return to Step 4 at most $h^e(x)$ many times, and will eventually wait forever at Step 4 (or instead wait forever at Step 2, in which case $h^e(x)$ is partial). Thus T^e_k will not trace $f^A(x)$, and the requirement will be satisfied.

Towards a full strategy for P_k^e. The complication comes from the fact that enumerating an element into A requires us to pay a cost; specifically, if z is the element enumerated, and s is the current stage, we must pay $c(z,s)$. Since we must ensure that $\sum_s c(A_s) < \infty$, we must be careful that changes on behalf of P_k^e contribute only a small amount.

Suppose that for some $c > 1$ and some $n > 0$, P_k^e has claimed for its future use $2^{c \cdot n}$ many elements of B^e, and that on all of these elements, h^e takes value at most n. P_k^e will follow the basic strategy on the first of these elements, but will discard this and choose the next one if the cost of clearing the computation rises too high. What is "too high"? This will depend on how much progress P_k^e has made on the current witness. Initially (if P_k^e has not yet cleared any computations on this witness), P_k^e will discard the witness if the cost of clearing the computation rises beyond $2^{-c \cdot n}$. Each time P_k^e completes a loop of the basic strategy (that is, each time it clears a computation), its threshold is multiplied by 2^{c-1}. So if the computation on the witness has been cleared m times, P_k^e will discard the witness if the cost of clearing the computation rises beyond $2^{-c \cdot n + (c-1) \cdot m}$. When it discards a witness and begins working with the next one, P_k^e's threshold returns to $2^{-c \cdot n}$.

We will perform the calculations later, but following the above strategy, the total cost of changes made on behalf of P_k^e will turn out to be bounded by $2^{-n+1} + 2^{-c+2}$. So we arrange to keep this value small.

Full strategy for P_k^e. We merely formalize the above. To assist, we keep functions $i_k^e(s)$ and $m_k^e(i,s)$, which track the current witness x_i being worked with and the number of times we have redefined the function at that witness, respectively. Below, s is always the current stage. When not otherwise specified, we always define $i_k^e(s+1) = i_k^e(s)$ and $m_k^e(i, s+1) = m_k^e(i,s)$.

(1) Choose $c, N \geqslant k + e + 3$.
(2) Search for an $n > N$ and $2^{c \cdot n}$ many elements $x_1, \ldots, x_{2^{c \cdot n}} \in B^e$ with $x_i > N$ and $h^e(x_i)\downarrow \leqslant n$ for all $1 \leqslant i \leqslant 2^{c \cdot n}$.
(3) Define $i_k^e(s) = 1$.
(4) Define $m_k^e(i_k^e(s), s) = 0$.
(5) Choose a large $z > s$. Define $f^A(x_{i_k^e})[s] = s$ with use $A_s \upharpoonright_{z+1}$.
(6) Wait for a stage $s > z$ when $f^A(x_{i_k^e})[s] \in T_k^e(x_{i_k^e})[s]$. While waiting, if $f^A(x_{i_k^e})[s]$ becomes undefined, redefine it to the same value with the same use.
(7) If $c(z,s) > 2^{-c \cdot n + (c-1) \cdot m_k^e(i_k^e(s), s)}$, define $i_k^e(s+1) = i_k^e(s) + 1$ and return to Step 4. Otherwise, enumerate z into A_{s+1}, define $m_k^e(i_k^e(s), s+1) = m_k^e(i_k^e(s), s) + 1$ and return to Step 5.

Organizing the construction. The strategies mostly act in isolation, with the following exception: we do not act for P_{k+1}^e until the strategy for P_k^e has chosen its elements x_i. We then require that the N chosen by the strategy for P_{k+1}^e is larger than any of the x_i chosen by the strategy for P_k^e. This is only to prevent interference between strategies.

Verification. We proceed as a sequence of claims.

Claim 8.1. *Fix e, k. Let c and n be the values chosen by the P_k^e-strategy, and let s be any stage. Then $i_k^e(s) \leqslant 2^{c \cdot n}$, and*

$$\sum_{i=1}^{i_k^e(s)-1} 2^{-c \cdot n + (c-1) \cdot m_k^e(i,s)} < 1.$$

Proof. For every $1 \leqslant i < i_k^e(s)$, let $s_i < s$ be the final stage with $i_k^e(s_i) = i$, and let z_i be the use of the $f^A(x_i)[s_i]$ computation. By construction, $z_i < s_i < z_{i+1}$, $m_k^e(i,s) = m_k^e(i,s_i)$ and $c(z_i, s_i) > 2^{-c \cdot n + (c-1) \cdot m_k^e(i,s_i)}$. So

$$\sum_{i=1}^{i_k^e(s)-1} 2^{-c \cdot n + (c-1) \cdot m_k^e(i,s)} < \sum_{i=1}^{i_k^e(s)-1} c(z_i, s_i) < 1$$

by Observation 7.

Further, since

$$\sum_{i=1}^{i_k^e(s)-1} 2^{-c \cdot n + (c-1) \cdot m_k^e(i,s)} \geqslant \sum_{i=1}^{i_k^e(s)-1} 2^{-c \cdot n} = (i_k^e(s) - 1) \cdot 2^{-c \cdot n},$$

it follows that $i_k^e(s) - 1 < 2^{c \cdot n}$, and thus $i_k^e(s) \leqslant 2^{c \cdot n}$. □

Claim 8.2. *Fix e, k. Let c and n be the values chosen by the P_k^e-strategy. Then the total cost of elements enumerated by the P_k^e-strategy is at most 2^{-k-e}.*

Proof. Let $\ell = \max_s(i_k^e(s))$, and for all $i \leqslant \ell$, let $m_k^e(i) = \max_s m_k^e(i,s)$. Fix $1 \leqslant i < \ell$, and let us first consider those stages s with $i_k^e(s) = i$. Let z_s be an element enumerated by the P_k^e-strategy at such a stage s. By construction, $c(z_s, s) < 2^{-c \cdot n + (c-1) \cdot m_k^e(i,s)}$. So the total cost of elements enumerated by the P_k^e-strategy at such stages s is at most

$$\sum_{m=0}^{m_k^e(i)-1} 2^{-c \cdot n + (c-1) \cdot m} \leqslant 2^{-c \cdot n + (c-1) \cdot m_k^e(i) - c + 2}.$$

Now, summing over all all $1 \leqslant i < \ell$, we get

$$\sum_{i=1}^{\ell-1} 2^{-c \cdot n + (c-1) \cdot m_k^e(i) - c + 2} = 2^{-c+2} \sum_{i=1}^{\ell-1} 2^{-c \cdot n + (c-1) \cdot m_k^e(i)} < 2^{-c+2}$$

by Claim 8.1.

Now it remains to consider those stages s with $i_k^e(s) = \ell$. Because $h^e(x_\ell) \leqslant n$, we know that $m_k^e(\ell) \leqslant n$. So the cost of elements enumerated by the P_k^e-strategy at such stages s is at most

$$\sum_{m=0}^{n-1} 2^{-c \cdot n + (c-1) \cdot m} \leqslant 2^{-c \cdot n + (c-1) \cdot n + 1} = 2^{-n+1}.$$

So the total cost of all elements enumerated by the P_k^e-strategy is at most $2^{-n+1} + 2^{-c+2}$, which, by choice of c and N, and since $n > N$, is at most 2^{-k-e}. □

Claim 8.3. *A is K-trivial.*

Proof. Since every element enumerated into A is enumerated by one of the P^e_k-strategies, we can bound $\sum_s c(A_s)$ by summing over the individual strategies. By the previous claim, this becomes $\sum_e \sum_k 2^{-k-e} = \sum_e 2^{-e+1} = 4 < \infty$. By Lemma 6, this implies A is K-trivial. □

Claim 8.4. *If h^e is a total computable order which obeys the hypothesis of the theorem, then we eventually begin acting for every P^e_{k+1}-strategy. That is, every P^e_k-strategy eventually finds its elements $x_1, \dots, x_{2^{c\cdot n}} \in B^e$.*

Proof. Proof by induction on k. Assume that we have begun acting for the P^e_k-strategy, and have chosen appropriate values c and N. By Lemma 4, there is some n with $h^e(N + 2^{(c+e+2)\cdot n}) \leqslant n$. Since h^e is an order, for every $x \in [N+1, N+2^{(c+e+2)\cdot n}]$ we know $h^e(x) \leqslant n$. The intersection of this interval with B^e contains at least $2^{c\cdot n}$ many elements, so these are sufficiently many elements for the strategy to select. Thus the strategy will eventually find its desired elements, and so we will begin acting for the P^e_{k+1}-strategy. □

Claim 8.5. *If h is a total computable order with $\sum_{x=0}^{\infty} 2^{-d\cdot h(x)} = \infty$ for all $d > 0$, then A is not jump traceable at order h.*

Proof. Suppose not, and fix an h-bounded c.e. trace $\langle T(x)\rangle_{x\in\omega}$ tracing f^A. Fix e and k with $h = h^e$ and $\langle T(x)\rangle_{x\in w} = \langle T^e_k(x)\rangle_{x\in\omega}$. By the previous claim, we eventually begin acting for the P^e_k-strategy, and this strategy eventually selects its elements at Step 2. By Claim 8.1, $\ell = \lim_s i^e_k(s) \leqslant 2^{c\cdot n}$.

Let s_0 be the least s with $i^e_k(s) = \ell$. By choice of s_0, this is the final stage at which the P^e_k-strategy returns to Step 4. Since $h^e(x_\ell) \leqslant n$, we know that $m^e_k(\ell, s) \leqslant n$ for every stage s, and so after stage s_0, the P^e_k-strategy returns to Step 5 only finitely many times. So in the limit, the P^e_k-strategy must wait at Step 6 forever, and thus $f^A(x_\ell)$ is defined and not contained in $T^e_k(x_\ell)$. □

This completes the proof. □

References

[1] Laurent Bienvenu, Rod Downey, Noam Greenberg, Andre Nies, and Daniel Turetsky. Characterizing lowness for Demuth randomness. *Submitted.*

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

[3] Rodney G. Downey and Denis R. Hirschfeldt. *Algorithmic randomness and complexity.* Theory and Applications of Computability. Springer, New York, 2010.

[4] Rupert Hölzl, Thorsten Kräling, and Wolfgang Merkle. Time-bounded Kolmogorov complexity and Solovay functions. In *Mathematical foundations of computer science 2009*, volume 5734 of *Lecture Notes in Comput. Sci.*, pages 392–402. Springer, Berlin, 2009.

[5] 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.

[6] André Nies. Calculate of cost functions. *In preparation.*

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

[8] André Nies. *Computability and randomness*, volume 51 of *Oxford Logic Guides.* Oxford University Press, Oxford, 2009.

[9] Sebastiaan A. Terwijn and Domenico Zambella. Computational randomness and lowness. *J. Symbolic Logic*, 66(3):1199–1205, 2001.

E-mail address: dan@msor.vuw.ac.nz

Department of Mathematics, Victoria University of Wellington, Wellington, NEW ZEALAND