THE UNIVERSITY OF AUCKLAND
NEW ZEALAND

DEPARTMENT OF MATHEMATICS
DEPARTMENT OF COMPUTER SCIENCE

RANDOMNESS, TRACEABILITY AND HIGHNESS NOTIONS

Author:
Jingwei Zheng

Supervisor:
Prof. A.Nies

Submitted in fulfillment of the requirements of the degree of MSc in Mathematics. 28 February 2013. Last update 21 February 2013.

RANDOMNESS, TRACEABILITY, AND HIGHNESS NOTIONS

Contents

This thesis is dedicated to my loving mother, who supported me through my studies.

1. Introduction

"Scientists use chance, or randomness, to mean that when physical causes can result in any of several outcomes, we cannot predict what the outcome will be in any particular case."

-Douglas Futuyma

1.1. Algorithmic randomness.

The area of algorithmic randomness has been widely studied in past decades amongst recursion theorists. The area revolves around studying infinite strings and determining which class, if any, of randomness notions they belong to.

There were some remarkable but unsuccessful attempts to define algorithmic randomness during the early to mid 20th century. The original approaches were later found to be not quite strong enough to capture the notion. Researchers like Richard von Mises tried to characterise randomness in [30] with statistical tests like the law of large numbers. However, observe the following sequence:

10101010101010101010101010101010101010101010.....

This passes the law of large numbers, since in the limit, the number of 0s and 1s will be equal. However, one does not need to get into mathematical definitions to realise that the sequence is far from random.

Following the initial attempt, Von Mises included numerous more rules in his characterization. These were all later proven to be void by Wald in his paper [1]. He showed that having a countable set of rules is not sufficient. Its only later with the rise of computability theory did the choice of selection rules become more clear.

The key breakthrough came from mathematician Per Martin-Löf, he combined the idea of tests with the theory of computation in the 1960s [18]. This contrasts with the idea of randomness in probability theory. He proposed that a sequence is random if it passes a infinite number of tests. Each of these tests being a property that a non-random set would have. Hence, if it exhibits none of these properties, it must be random.

If a sequence is random in this way, we can conclude that it does not exhibit any special properties. For example, the binary representation of the irrational constant e will not be in the set of elements that is Martin-Löf random, nor will any sequence where there is always a run of 0s after a run of 1s.

There are two other equivalent approaches to defining Martin-Löf randomness, the incompressibility approach and the martingale approach. A sequence is incompressible if it can not be represented by a shorter sequence in the system. The martingale approach uses a series of betting strategies, and can be thought of as a game where no matter how you bet, the outcome will always be that you break even in the limit.

Since both of these notion are not used in the thesis, we shall mostly ignore them and instead refer solely to the testing approach.

After the first formulation, many other notions of randomness arose, however, Martin-Löf randomness still remains the most recognised notion. To the extent that sometimes, the term "random" referring to a sequence without clarification is automatically taken to mean Martin-Löf randomness.

1.2. **Traceability.**

The group of combinatorial properties known as the class of tracing notions, is a relatively more intuitive notion. Little work was done in this area until the late 1990s. Ishmukhametov, Terwijn and Zambella were the first to truly study the notion in [29]. They discovered its importance in classical recursion theory as well as algorithmic randomness. One of the main motivation for studying the area during the early stages was its strong link to array computability and hyperimmune-freeness [25].

Intuitively speaking, the basic traceability property on a function f dictates that $f(x)$ is somewhat effectively recoverable from a bounded finite set T_x, called its trace. The bounds imposed on T_x is to ensure we do not just "wildly guess", so that there is some form of logic in enumerating into T_x and we are concerned with "resource placement". For one would agree that without the bound, we can simply allow T_x to contain the enumeration of the entire natural numbers, and eventually, no matter what f is, $f(x)$ will be included. However, sometimes the bound may be varied. We will see later, for total functions, the bound is interchangeable as long as it exhibits some specific property (namely, being an order function).

This idea of tracing discrete functions is similar to the idea of estimating continuous functions in analysis. Instead of a ϵ that bounds the error, we have a set of possibilities contained in T_x. Just like how the error can not exceed ϵ, our $f(x)$ has to definitely be in T_x.

One major difference between the two is that, in analysis, the estimates are usually in a sense "close" to each other, but T_x can contain elements that are no where near each other in terms of size. For example, when enumerating T_x, we may simulate $f(x)$ with multiple functions, $h_1, h_2, h_3, \ldots$. Now T_x will contains values from each $h_n(x)$, it could be that h_1 is the function 2^x, h_2 the function x^x and so on. The end result will be that T_x contains some values that are extremely large, and some very small. If this were a real life prediction, it would be not very good, even though the truth is in T_x somewhere, the variation is way too much. However, from a mathematical sense, that does not matter, since we only care about if we used our resource properly and not overloaded T_x with elements, the variation is unimportant.

Although the concept may seem intuitive, there are many different traceability classes that have been introduced since its discovery. These notions include c.e. traceability [28], jump traceability [20], computable traceability [29] and strong jump traceability [13]. We focus our study on the first two in this thesis, and investigate some of their variants in relation to randomness. Of all the traceability notions, c.e. traceability was arguable the first to be introduced. It is also the most simple: the only condition for a function to be c.e. traceable is that it can be traced by some bounded uniformly computably enumerable $\{T_x\}_{x\in\mathbb{N}}$. Later, it is also shown that c.e. traceability is the same as array computability (every function computable from a set A is dominated by some ω-c.a. function) for c.e. sets. This sparked a lot of researcher's interest in the field, since array computability was a well studied subject back then.

Some other ways of looking at traceability that had recently been included are ω-c.e. traceability and Δ^0_2 traceability [11]. They are essentially the sets that are powerful enough to trace all ω-c.a. and Δ^0_2 functions respectively when used as an oracle. Both these variants are focal points in section 3 and section 6.

In this thesis, we employ several highness principles to build a bridge between traceability and its relations to randomness. On the surface, one can argue that algorithmic randomness, a notion that deals with randomness of strings, and traceability, a notion of generating approximations, seem to have no real deep interplay. However, this thesis makes multiple advances on the relationship of the two.

1.3. **Outline.**

In Section 2, we first give formal definitions for ideas that are vaguely defined here and offer some motivation for the rest of the thesis. We lay down groundwork by showing elementary results that are going to be useful throughout the thesis.

In Section 3 we get to work on demonstrating what exactly ω-c.e.-tracing sets are capable of. There we show even though being ω-c.a. is a highness property, they can still be low in the usual sense. Furthermore, they are not all high for that matter. We ask questions like "can they be close to computable? How far are they from $\emptyset'$?".

The short Section 4 is there to lay some ground work for Schnorr randomness relative to $\emptyset'$. We explore equivalences between Schnorr randomness, the relatively new notion of limit random and Martin-Löf random sets. We characterize Schnorr randomness relative to $\emptyset'$ via Martin-Löf and limit randomness, then use this ground work in Section 6 to give simpler alternate proofs to theorems regarding highness.

Section 5 revolves around interactions between K-triviality (a set is K-trivial if it is computable from $\emptyset$ in a Kolmogorov way), Demuth randomness and super-lowness. We demonstrate which of the three notions is the most powerful and give a short display of the usefulness of the High($\mathcal{C}$, $\mathcal{D}$) property.

Then, we start making some links between randomness and traceability in section 6. We first connect Δ^0_2 and ω-c.a.-tracing with Demuth and Schnorr randomness respectively. In the latter half of the section, a new traceability notion was introduced: Demuth traceability. We ask how much cupability power it has and how closely related is it to ω-c.e. traceable sets. The answer was somewhat surprising to me, but after moments thought, it became natural.

At last, in chapter 7 we ask some questions about the interchangeability of the trace bound. Can anything other than c.e.-traceable sets have this property? How many conditions do we have to apply to our function to be able to achieve this result? Can the techniques in Section 6 be used in any other traceability variants? Though not all these questions are fully answered, we give at least some satisfactory results. We conclude this section by summing up anything unanswered and leave it for future brave researchers that wished to tackle the problems that bested myself.

2. **Preliminaries**

In this section, we give some frequently used definitions and notations. Most of these are standard computability theory definitions. Readers with experience in recursion theory should be fairly familiar with them. To make sure we do not deviate too much from the norm when choosing the symbols for our notations, this thesis will closely follow a similar style of writing as Nies book [21].

2.1. **Notation and conventions.**

Most of our notations, conventions and terminologies will be introduced when needed. Here we give a few non-technical terms and ideas that are frequently referred to throughout the thesis.

The model of computation, as always, is a Turing Machine. Though not frequently referred to, we view computability as based on the acceptance of the Church-Turing Thesis.

As usual we mostly use the Greek letter φ and Φ to denote functions for construction proofs. Likewise W_e will represent the domain of the eth computable function, we also use the subscript notation to denote a function or set during stage s, i.e. $W_{e,s}$ will mean W_e during the sth stage of the construction.

The capital letters A, Z and U, V are usually used to represent sets. A and U will be used the most often and Z and V is used only if we are attempting to build two sets in the same proof. A and Z will mostly be used in constructive proofs representing generic sets while U and V are used in proofs that revolves around building tests for randomness notions. The alternate styled $\mathcal{C}$ and $\mathcal{D}$ will also be used to refer to a class of notions or sets.

Naturally, there will be cases where we would like to view all relevant computations at some stage s. Instead of using the subscript notation on everything, we denote $[s]$ to mean all computation is at stage s. For example, $\Phi_e^A[s] = \Phi_{e,s}^{A_s}$ and $W_e^A[s] = W_{e,s}^{A_s}$.

All computations are assumed to be discrete, therefore all computations yield natural numbers unless stated otherwise. The topological space we will refer to throughout the thesis is Cantor space, i.e. the topological space that is homeomorphic to the cantor set, denoted 2^ω or $2^{\mathbb{N}}$.

All strings are defined to be binary strings. We denote the set of finite binary strings by $2^{<\omega}$, the empty string by λ or $\varnothing$, the length of a string τ by $|\tau|$, the open cylinder generated by a σ in $2^{<\omega}$ by $[\sigma]$. We usually refer to cylinders when we are dealing with a binary tree. The concatenation of two strings τ and σ will be denoted by $\tau^\frown\sigma$ and we will write $\tau(n)$ to represent the nth value of a string τ.

For the lack of a better symbol, we shall abuse mathematical notations and denote the uniform measure on Cantor space by λ as well. This should not cause any confusion as if both empty strings and the uniform measure are required, the context should be clear as to which one we mean. In the worst case scenario when disguising the two is impossible, we denote the empty string by the alternative notation, $\varnothing$.

The lexicographic ordering on $2^{<\omega}$, $<_{\text{Lex}}$ is an binary relation between two strings τ and σ. We say that $\tau <_{\text{Lex}} \sigma$ if either $|\tau| < |\sigma|$ or $|\tau| = |\sigma|$ but $\tau(n) = 0$ for the least n such that $\tau(n) \neq \sigma(n)$. This will be the primary ordering used in the thesis, since it is a great way to present an ordering system on a binary tree.

All encoding, unless stated otherwise, are recognised to be some form of prime factorization. Gödel encoding would make sense most of the time. That is, given a sequence of n-tuples $(x_1, x_2, x_3, x_4..., x_n)$ of positive integers, the Gödel encoding of the sequnece is the product of the first n primes raised to their corresponding values in the sequence:

$$\text{enc}(x_1, x_2, x_3, x_4..., x_n) = 2^{x_1} \times 3^{x_2} \times 5^{x_3}....p_n^{x_n}.$$

This encoding system is used to ensure the uniqueness of our encryption: by the fundamental theorem of arithmetic, every positive integer has a unique prime factorization.

We shorthand the phrases computably enumerable with c.e., the phrase recursively enumerable with r.e. and computably approximation by c.a. Note that computable enumerable and computably approximation are different terms: c.e. is used when referring to sets and classes while c.a. is used when we are referring to functions.

2.2. **Basic definitions.**

We assume familiarity with most of the following definitions, all but a few were introduced during the early development stages of recursion theory. Hence they will not be explained in too much details.

Definition 2.1. Given $A \subseteq \omega$, the Turing jump of A is defined to be the set

$$A' = \{e \in \omega : \Phi_e^A(e) \downarrow\},$$

we will also denote the in computable halting problem by:

$$\emptyset' = \{x : \Phi_x(x) \downarrow\},$$

or if one prefers the Turing equivalent two variable version:

$$\emptyset' = \{\langle x, y\rangle : \Phi_x(y) \downarrow\}$$

Definition 2.2. [16, Kolmogorov, 1965]. Let $f : 2^{<\omega} \to 2^{<\omega}$ be a partial computable function. We call the Kolmogorov complexity of a string σ with respect to f

$$C_f(\sigma)\text{=}\min\{|\tau| : f(\tau) = \sigma\},$$

If the set is empty, for convention sake, we will define the minimum string length to be infinite, i.e. ∞ . We say that a string σ is random relative to f if $C_f(\sigma) \geq |\sigma|$.

The function f that one chooses is not very important. As Kolmogorov showed that there is a universal function f for the descriptive system C. This is in the sense that f can act as any other function within C for a increase in constant lengths.

Furthermore, we define the prefix-free Kolmogorov complexity of a string σ to be:

$$K_{\mathcal{T}}(\sigma) = C_{\mathcal{U}}(\sigma)$$

where $\mathcal{U}$ is the universal prefix-free machine, that is, it has a prefix-free domain.

Definition 2.3. [18, Martin-Löf, 1966]. A Martin-Löf test is a uniform sequence $\{U_n\}_{n\in\mathbb{N}}$ of c.e. classes such that for each $n \in \mathbb{N}$, $\lambda(U_n) \leq 2^{-n}$. A set $A \subseteq \mathbb{N}$ fails the test if $A \in \bigcap_n U_n$ and we call A a Martin-Löf null, otherwise, we say Z passes the test. Furthermore, A is Martin-Löf random if it passes all Martin-Löf tests. The class of all Martin-Löf random sets are denoted by MLR.

This notion of randomness is used to help us distinguish random infinite sequences from those that are not.

As mentioned in the introduction, there are two other equivalent approaches to defining Martin-Löf randomness, the incompressibility approach and the martingale approach. We give a short formal description of the two here, but do not go into too much details since we are working with the above definition for the sake of this thesis.

The incompressibility approach measures was introduced by Levin and Chaitin. They originally hoped that if A is random, then $C(A \upharpoonright n) \geq n - O(1)$. Sadly, this is impossible and no set satisfies this definition. They then abandoned the idea of characterization via the plain complexity C and instead used K, the prefix-free Kolmogorov complexity. This yielded a much better result for Levin in his follow up paper [17]. There, he arrived at the following definition:

Definition 2.4. [23, Schnorr 1971]. A set A is 1-*random* if $K(A \upharpoonright n) \geq n - O(1)$.

As the name suggests, there are more "levels" of randomness which we will not go into. An informal way to look at this definition is that A considered random if you can not represent it with a shorter sequence in the system. So to retrieve the information that A holds in the first n bits, you have to have at least n bits of information available, and nothing less. One can probably now see in a non formal level that this is somewhat similar to Martin-Löf randomness. Since exhibiting no special properties makes a sequence hard to compress.

On the other hand, a martingale is similar to gambling against a casino that is just trying to break even:

Definition 2.5. a function $f : 2^\omega \to \mathbb{R}^{\leq 0}$ is a martingale if for all σ

$$f(\sigma) = \frac{f(\sigma 0)+f(\sigma 1)}{2}$$

A martingale f is successful on a set A if $\lim\sup_n f(A \upharpoonright n) = \infty$.

Schnorr [24] showed that a set is 1-random iff no c.e martingale is successful on it. The idea of a martingale is that we have a starting amount of capital (money) $d(\sigma)$ and we are betting on the outcome of the nth bit of σ with some strategy. The game itself is guaranteed to be fair (you can think of it as betting odds or evens on a roulette table in a casino where the

0s and 00s has been taken out). In other words, no matter what strategy you choose to employ or what strategy is used again you, the expected value after some observation is the same as the value before the observation.

There are many other notions of randomness used in this thesis, like Schnorr randomness and Demuth randomness. We present their definitions and unveil some interesting and useful facts about Schnorr randomness.

Definition 2.6. [7, Demuth,1988]. A Demuth test is a uniform sequence $\{U_n\}_{n\in\mathbb{N}}$ of c.e. classes such that for each $n \in \mathbb{N}$, $\lambda(U_n) \leq 2^{-n}$, and there is a function $f \leq_{wtt} \emptyset'$ such that $U_n = [W_{f(m)}]^{\prec}$. A set A passes the Demuth test if $A \notin U_n$ for all but finitely many n. Furthermore, A is Demuth random if it passes all Demuth tests.

Definition 2.7. [23, Schnorr, 1971]. A Schnorr test is a uniform sequence $\{U_n\}_{n\in\mathbb{N}}$ of c.e. classes such that for each $n \in \mathbb{N}$, $\lambda(U_n)$ is uniformly computable. A set $A \subseteq \mathbb{N}$ fails the test if $A \in \bigcap_n U_n$, otherwise we say Z passes the test. Furthermore, A is Schnorr random if it passes all Schnorr tests. The class of all Schnorr random sets are denoted by SR.

We define "weaker" tests because sometimes the pass conditions on Martin-Löf randomness is too strict, and we have to "losen" the pass conditions to get to our desired result. In this thesis, we work with Schnorr tests as if they were a Martin-Löf tests $\{U_n\}_{n\in\mathbb{N}}$ with $\lambda(U_n) = 2^{-n}$. In the next subsection, we will show that these definitions indeed are interchangeable.

Definition 2.8. [27, Solovay, 1975] A Solovay test is a sequence $\{S_n\}_{n\in\mathbb{N}}$ of uniformly c.e. classes such that $\sum \lambda(S_n) < \infty$. A set A passes this test if it is in only finitely many of the S_n.

In the next subsection, we will show that this notion is equivalent to Martin-Löf randomness. Downey chose to extend this definition into a notion that coincides with Schnorr randomness:
Definition 2.9. [9, Downey and Griffiths, Definition 6.1.9]. A total Solovay test is a sequence $\{S_n\}_{n\in\mathbb{N}}$ of uniformly c.e. classes such that $\sum \lambda(S_n) < \infty$ and is a computable real. A set A passes this test if it is in only finitely many of the S_n.

Now we state the formal definition for the notion of traceability that this thesis focuses on:

Definition 2.10. [32, Zambella,1990]. Let h be an order function, that is, an unbounded, non-decreasing, non-negative, computable function. We say that g is traceable with bound h if there is a sequence $(T_n)_{n\in\mathbb{N}}$ of non-empty sets such that $g(n) \in T_n$ and $|T_n| \leq h(n)$ for each n. Furthermore, we call f c.e. traceable if $(T_n)_{n\in\mathbb{N}}$ is uniformly c.e. i.e. $T_n = W_{f(n)}$ for all n.

The following (somewhat weak) highness property was introduced by Greenberg and Nies [15]; it coincides with the class $\mathcal{G}$ in [21, Proof of 8.5.17].

Definition 2.11. A set A is *ω-c.a.-tracing* if each function $f \leq_{\text{wtt}} \emptyset'$ has a A-c.e. trace $(T^A_x)_{x\in\mathbb{N}}$ such that $|T^A_x| \leq 2^x$ for each x.

A stronger condition is that every Δ^0_2 function f must be traced:

Definition 2.12. A set A is Δ^0_2-*tracing* if each Δ^0_2 function f has a A-c.e. trace $(T^A_x)_{x\in\mathbb{N}}$ such that $|T^A_x| \leq 2^x$ for each x. (One can also say that $\emptyset'$ is c.e. traceable *by* A.)

For more background on tracing see [21, Sections 8.2,8.4].

2.3. Important earlier results.

In this section we give a few earlier results. Some of these results will be referred to frequently throughout the thesis, others are there to convince us that there indeed is a very interesting connection between some of these different notions. We first dive into proving the results promised in the last section:

Theorem 2.13. [24, Schnorr, 1971]. *Let $\{U\}_{n\in\mathbb{N}}$ be a Schnorr test. There exists a Schnorr test $\{V_n\}_{n\in\mathbb{N}}$ such that $\lambda(V_n) = 2^{-n}$ and $\bigcap_n U_n \subseteq \bigcap_n V_n$.*

Proof. For each U_n, we effectively enumerate a V_n as follows:

First, for each n, we compute a sequence of rationals $r_o < r_1 < \ldots < \lambda(U_n)$ such that $\lambda(U_n) \leq 2^{-i} - r_i$.

Second, enumerate into V_n in stages:

At stage s, first check if $V_n[s] \supseteq U_n[s]$. If not, put all elements in $U_n[s] - V_n[s]$ into $V_n[s]$.

Now, select i such that $\lambda(V_n[s]) < 2^{-n} - 2^{-i}$ and $\lambda(U_n[s]) \geq r_i$, add open sets to $V_n[s]$ to ensure its measure is equal to $2^{-n} - 2^{-i}$. At the end of the computation, we have $V_n \supseteq U_n$ since at each stage $V_n[s] \supseteq U_n[s]$, $\lambda(V_n) = 2^{-n}$ as 2^i must eventually shrink to 0 when r_i approaches $\lambda(U_n)$. □

One of the most attractive properties of Martin-Löf randomness the existence of an universal machine, i.e. a test that contains all Martin-Löf nulls set. This allows us to work with just one sequence of Martin-Löf test.

Theorem 2.14. [18, Martin-Löf, 1966]. *There exists a universal Martin-Löf test.*

Proof. We build such a test via the method of selective listing.

First, effectively list each c.e. subsets of $2^{<\omega}$, $\{S^0_n\}_{n\in\mathbb{N}}$,$\{S^1_n\}_{n\in\mathbb{N}}$..., we use this to build a list of all Martin-Löf tests.

Next, we make sure that each $\{S^i_n\}$ contribute the correct amount of measure:

Enumerate S^i_n until the measure of $[S^i_n]$ is about to exceed 2^{-n}, i.e. if at some s stage of the enumeration $\lambda([S^i_n])$ becomes greater than 2^{-n}, cancel all elements enumerated during stage s and stop the enumeration. We then build $\{R^i_n\}_{n\in\mathbb{N}}$ by setting $R^i_n = S^i_n$. Note that $\{R^0_n\}_{n\in\mathbb{N}}$,$\{R^1_n\}_{n\in\mathbb{N}}$,$\{R^2_n\}_{n\in\mathbb{N}}$... is indeed an effective listing of all Martin-Löf tests.

Now, set $U_n = \bigcup_i [R^i_{n+1+i}]$. Since each R^i_n is a Martin-Löf test, the U_n are uniformly Σ^0_1.

Lastly, we check if the measure is correct. As by our enumeration:

$$\lambda(U_n) = \textstyle\sum_i \lambda([R^i_{n+1+i}]) \leq \sum_i 2^{-(n+i+1)} = 2^{-n}.$$

So $\{U_n\}_{n\in\mathbb{N}}$ is a Martin-Löf test and for any other Martin-Löf test $\{V_n\}_{n\in\mathbb{N}}$ we have an i such that $V_n = [R^i_n]$ for each n. Then $V_{n+1+i} \in U_n$ and so V_n is in some member of $\{U_n\}_{n\in\mathbb{N}}$ for all but finitely many n. Hence $\bigcap_n V_n \in \bigcap_n U_n$.

□

Another important property of Martin-Löf's definition is its equivalent characterization via Solovay tests.

Theorem 2.15. [27, Solovay, 1975]. *A set is Solovay random iff it is Martin-Löf random.*

Proof. Each individual U_n in a Martin-Löf test has measure 2^{-n} and the sum $\sum_n \lambda(U_n)$ equals $\sum_n 2^{-n}$ is a geometric sequence that converges absolutely to 1. So a Martin-Löf test is a Solovay test.

Now suppose that A is Martin-Löf random and $\{S_n\}_{n\in\mathbb{N}}$ is a Solovay test. Since $\sum_n \lambda(S_n) < \infty$, there exists m such that $\sum_{n>m} \lambda(S_n) < 1$.

Define $U_k = [\{\sigma : \exists^{\leq 2^k} n > m \text{ and } [\sigma] \in S_n\}]$. Each U_k has at most measure $2^{-k} \times \sum_{n>m} \lambda(S_n)$, and since $\sum_{n>m} \lambda(S_n) < 1$ we have $\lambda(U_k) < 2^{-k}$. Thus $(U_k)_{k\in\mathbb{N}}$ is indeed a Martin-Löf test.

Since A is Martin-Löf random there must a a U_k such that $A \notin U_k$. Since there are only m Solovay test components unaccounted for by $(U_k)_{k\in\mathbb{N}}$, A is in at most $2^k + m$ many S_n. Hence, A is Solovay random. □

Schnorr randomness also have an attractive parallel property, in the sense that it can be characterized via *total* Solovay tests.

Theorem 2.16. [8, Downey, R. and Griffiths, E, 2004]. *A set is Schnorr random iff it passes all total Solovay tests.*

Similar to the idea of a Universal Turing Machine, one would ask if there is some form of similar function that in a way is interchangeable between different traceable sets. In fact, one of the earliest results in the field was that, if we trace only total functions, the Tewijn-Zambella construction (see [21, Thm. 8.2.3]) allows us to change the bound 2^x to any order function without changing the tracing property.

Theorem 2.17. *A c.e. traceable set is c.e. traceable via every order function.*

Proof. Let A be c.e. traceable with some order function h and we are given some other order function q.

First define a computable function $r(n)$ in terms of inputs in q that is strictly less than $h(n+1)$:

$$r(n) = 1 + \max\{i \colon q(i) < h(n+1)\}$$

For example, if we have $h(n) = n+1$ and $q(i) = i^2$, then $r(n) = \sqrt{n+1}$. You can recognise r as an indirect bound function represented by the order functions h and q.

Next we provide a trace $(T_i)_{i\in\mathbb{N}}$ with bound q for f. The idea is to remove some of the "guess work" required while building these trace sets.

Suppose f is total and $f \leq_T A$. Instead of working directly with f we instead encode it, let $g(n) = f\upharpoonright_{r(n)} = \text{enc}(0, 1, 2 ... r(n))$ as defined in section 2.2. Notice g and f are Turing equivalent since the encoding preserves reducibility. Now, we can modify $(T_i)_{i\in\mathbb{N}}$ accordingly to generate a c.e. trace $(T_n)_{n\in\mathbb{N}}$ for g with bound h. We can also assume that for each $\alpha \in T_n$

$$\alpha \in g \text{ and } |g^{-1}(\alpha)| = r(n)$$

or else the encoding does not properly define $f \restriction_{r(n)}$.

For each i, let $T_i = \{\alpha(i) : \alpha \in T_{n_i}\}$, where n_i is the least n such that $i\colon q(i) < h(n+1)$. By definition, we have $r(n_i) > i$, and since $|g^{-1}(\alpha)| = r(n) > i$, we have $f(i) \in T_i$ as well. Moreover, $|T_i| \leq |T_{n_i}|$ and $|T_{n_i}| \leq h(n_i)$. Since n_i is the least n such that $i\colon q(i) < h(n+1)$, we have $|T_i| \leq |T_{n_i}| \leq h(n_i) \leq q(i)$ for each i. □

It is also shown that the preceding argument can be adopted to a wide range of contexts where only total functions are traced in [21] section (8.2.29) and (8.2.15). Further, in section 6 of this thesis, we investigate this property in relations to some different functions.

One of the other remarkable theorems that this thesis will refer to is the characterization of K-trivial sets via Martin-Löf random sets.

Definition 2.18. We say A is K-trivial if $K(A \restriction n) \leq K(n) + c$ for some constant c. Where K is, again, the prefix-free Kolmogorov complexity.

It was proved by Nies using the "Golden Run" method. In [21, Section 5.4] he formally showed that:

Theorem 2.19. *The following are equivalent:*
(i) A is K-trivial.
(ii) A is low for Martin-Löf randomness.
(iii) There exists $X \leq_T A$ such that X is Martin-Löf random relative to A.

The proof also used work done by Downey, Hirschfeldt, Nies and Stephan in 2003 which showed that each K-trivial set is Turing incomplete. This result is crucial for proving the Day/Miller theorem in section 5.

3. **Traceability**

In this section, we focus our attention on ω-c.a.-tracing sets and their interaction with lowness. We prove multiple separation results that paints a clearer picture of where the ω-c.a.-tracing sets are in the arithmetical hierarchy. Since we are trying to build infinite sets that satisfies one or more condition, the main mathematical machinery used here will be injury arguments. Most of the questions answered in this section were originally set as open questions in the 2012 Logic Blog. The relationship between the two was first investigated in the 2010 paper [11]. The following result, Theorem 3.1 was one of the areas explored in the paper regarding the subject. It was shown that:

Fact 3.1. *No superlow set is ω-c.a.-tracing.*

Proof. The main idea of the proof is that we diagonalize against all the possible trace sets. Let Z be superlow, effectively list all Z-c.e. traces such that $|T^Z_{e,s}| \leq 2^x$ for each e, x. Since $Z' \leq_{wtt} \emptyset'$, $n \in T^Z_{e,x}$ iff there exists some ternary relation p such that $p(n, e, x) \notin Z'$. This relation is weak truth-table below $\emptyset'$. Now let $f(x)$ be the least number not in $\bigcup_{e \leq x} T^Z_{e,x}$. Since each $T^Z_{e,x}$ has a computable bound 2^x, the number of changes in f is also bounded by a computable function, say 2^x. Then $f \leq_{wtt} Z' \leq_{tt} \emptyset'$, and f is not traced by any Z-c.e. trace $(T^Z_x)_{x\in\mathbb{N}}$ such that $|T^Z_x| \leq 2^x$ for each x.

□

With some thought one can come to the conclusion that the above result comes very natural, since being ω-c.a.-tracing is a highness property. On the other hand, one can show that lowness is in fact possible. By [11, Journal version Cor. 25] we can see that:

Corollary 3.2. *There is an ω-c.a.-tracing low ML-random set.*

The result was derived directly from combining Theorem 11 and Theorem 23 of the same paper. Here, we use a construction method and build a low *c.e.* set instead.

Theorem 3.3. *Some low c.e. set A is ω-c.a.-tracing.*

Proof. We can not jump in straight away as we need to first obtain $(f_e)_{e\in\mathbb{N}}$, a list of all ω-c.a. functions to be traced. We do this by giving an approximation with reduction procedures and take the limit of (f_e) with respect to each procedure, here is a sketch of the procedure:

Let $\langle e \rangle$ be the e-th wtt reduction procedure (namely, e_0 indicates a Turing functional and e_1 is a computable bound on the use). At stage any s, we have an approximation $f_e(x)[s] = \langle e \rangle^{\emptyset'}(x)[s]$ and define this value to be 0 if this approximation is undefined. The function f_e is given by $f_e(x) = \lim_s \langle e \rangle^{\emptyset'}(x)[s]$.

The construction itself is similar to [26, Chapter 7, Theorem 1.1] on building low simple sets. We retain the same negative requirement and use a different approach to dealing with the issue of making our set ω-c.a.-tracing.

First we build a c.e. oracle trace $(T^A_x)_{x\in\mathbb{N}}$ with a fixed computable bound $h(x)$. We meet the postive requirement:

$$P_{e,x}\colon\ f_e(x) \in T^A_x\ (e \leq x).$$

We also meet the usual lowness requirements N_e.

$$N_e\colon\ (\exists^\infty_s \Phi^A_e(e)[s]\downarrow) \Rightarrow \Phi^A_e(e)\downarrow.$$

Fix an effective priority ordering of the requirements:
$N_0, P_{0,0}, N_1, P_{1,0}, P_{1,1}, N_2, P_{2,0}, P_{2,1}, P_{2,2}, N_3$.......

Strategy for $P_{e,x}$ at stage s: when there is a new value $y = f_e(x)[s]$, change A to remove the previous value (if any), unless its A-use is restrained by a stronger priority N-type requirement. Put $y \in T^A_x$ with large use on A.
Strategy for N_e at stage s: given A_s, set up restraint function for all e:

$$r(e,s) = \varphi^A_e(e)[s].$$

Where φ^A_e is the relative use function of Φ^A_e. If $\Phi^A_e(e)$ converges newly (by convention with use $\leq s$), restrain all weaker P requirements from changing $A\restriction_s$.

Construction of A
Stage $s = 0$ and A is empty.
Stage $s{+}1$. Given A_s, we have $r(e,s)$ for all e. We say $P_{i,x}$ *requires attention* if $f_i(x)[s-1] \neq f_i(x)[s] \downarrow= y$.
Find the least i requiring attention. If such a i exists, pick a fresh large number k, put $y \in T^A_x$ with use $k \in A$, i.e $A[k] = 1$. Then attempt to remove $f_i(x)[s-1] = y'$ from T^A_x (if it exists). As $y' \in T^A_x$, it must have a use k'. Set $A[k'] = 0$ if its not restrained by some stronger L requirement. We say that $P_{i,x}$ *receives attention.*
If no such i exists, do nothing, let $A_s = A_{s+1}$
Let $A = \bigcup_s A_s$. This ends the construction.
We say x *injures* N_e at stage $s+1$ if $x \in A_{s+1} - A_s$ and $x \leq r(e,s)$. Define the *injury set* for N_e,

$$I_e{=}\{x\colon\ x \text{ injures requirement } N_e \text{ at some stage } s+1\}.$$
$$= \{x\colon (\exists s)[x \in A_{s+1} - A_s \text{ and } x \leq r(e,s)]\}.$$

(The positive requirement are of course never injured.)

Claim 3.4. $(\forall e)[I_e$ *is bounded by some computable function* $g(x)]$.

Proof. Since each f_i is a ω-c.a. function, its number of changes is bounded by some computable function b_i. Hence each $P_{e,x}$ can change A at at most some $b_i(x)$ times. There are also only $((e{+}1)e)/2$ stronger P requirements that can injure N_e. Let $g(x) = \max\{b_i(x)\colon i \leq x\}{\times}(e(e{+}1))/2$. Then $|I_e| \leq g(x)$. ☐

Claim 3.5. *For every e, requirement N_e is met and $r(e) = lim_s r(e, s)$ exists.*

Proof. Fix e, by our first claim, choose a stage s_e such that N_e is not injured after stage s_e. If $\Phi^{A_s}_{e,s}(e)$ converges for $s > s_e$, then by induction on $t \geq s$, $r(e, s) = r(e, t)$ and $\Phi^{A_t}_{e,t}(e) = \Phi^{A_s}_{e,s}(e)$ for all $t \geq s$. So $A \upharpoonright r = A_s \upharpoonright r$. Hence $\Phi^A_e(e)$ is defined by the *use principle.* □

Claim 3.6. *For each e, x requirement $P_{e,x}$ is met.*

Proof. Fix i such that f_i is an ω-c.a. function. To meet $P_{i,x}$ we need to satisfy $f_i(x) \in T^A_x$. Claim 3.4, there is a stage s such that

$$(\forall t \geq s)(\forall e \leq i)\ [r(e, t) = r(e)].$$

Choose a stage $s' \geq s$ such that no stronger P requirements receives attention after stage s'. At stage s', assume $f_i(x)[s'] = y$ (if it has never received attention, assume $y = 0$), then $y \in T^A_x$. If $f_i(x)[s'] = f_i(x)[t]$ for all $t > s'$, we are done and $P_{e,x}$ is satisfied.

If not, $f_i(x)[s'] \neq f_i(x)[t] = y'$ for some t. We remove y from T^A_x (if allowed) and put $y' \in T^A_x$ with large use. Hence $P_{e,x}$ receives attentions at stage t, since $f_i(x)$ can only receive attention finitely often, as f_i is a ω-c.a function. Eventually at some stage $P_{e,x}$ will no longer receive attention and becomes met.

□

Claim 3.7. *There is a computable bound for $|T^A_x|$.*

Proof. Each time we put $f_e(x) \in T^A_x$ a *fresh large number* is used to impose a use on A. We may not remove $f_e(x)$ from T^A_x if a new restraint is raised by one of the stronger negative requirement N_i before f_e has changed its mind. Once a negative requirement raises a restraint, it may not change it until it is injured.

T^A_x contains $f_1(x), f_2(x).....f_x(x)$ as well as every approximated value leading up to each $f_i(x)$. Assuming each time a negative requirement raises a new restraint, each $f_i(x)$ can not leave T^A_x. By Claim 3.4 there is a computable bound $g(i)$ on how many times a N_i can be injured and there are only x many stronger N-type requirements. So $|T^A_x|$ is bounded by

$$x^2 \times \max\{g(i) : i \leq x\}.$$

□

By Claim 3.7 we have a computable bound for T^A_x, and the result of Terwijn and Zambella allows us to instead replace $h(x)$ with any other order function. Take this order function to be 2^x, then we have that A is ω-c.a.-tracing. □

3.1. The ω-c.a.-tracing sets can be close to computable.

We give another formal explication of the idea that an ω-c.e. set A can be high in one sense and low in another. Recall in the introduction section we described array computable degrees as degrees dominated by $\omega - c.a.$ functions, here we formally define the idea:
Definition 3.8. [10, 1996] A degree a is array computable if there exists a function $f \leq_{wtt} \emptyset'$ which dominates all functions $f \leq_T a$.

From here one could weaken the function f and arrive at the following:
Definition 3.9. [2, Definition 1.2] A degree a is weakly array computable if there exists a function $f \leq_T \emptyset'$ which dominates all functions $f \leq_T a$.

First note that A cannot be both Δ^0_2 and ω-c.a-tracing: in [2, Corollary 1.1], Barmpalias discovered that if a set is ω-c.a.-tracing then its not weakly array computable. We know that for c.e. sets, array computable was the same as as c.e. traceable. Relativising the corollary we can derive the following result:

Corollary 3.10. *ω-c.a.-tracing sets can be not Δ^0_2 tracing.*

This shows that being ω-c.a. and Δ^0_2 are indeed separate tracing notions. Furthermore, in [12, Rmk. 31] a question was asked in regards to array computability. In the paper, they questioned if there even exists an array computable ω-c.a.-tracing set. The following should answer the question very nicely. (Also note that the class of c.e. traceable sets contains all the superlow c.e. sets, but not all low c.e. sets. Thus this result and the previous result are independent.)

Theorem 3.11. *Some ω-c.a.-tracing c.e. set A is c.e. traceable.*

Proof. We will use a $\emptyset''$ tree construction. To make A c.e. traceable we meet requirements

$$S_e\colon\ \Phi^A_e \text{ total} \Rightarrow \text{build c.e. trace } (V_y)_{y\in\mathbb{N}} \text{ for } \Phi^A_e,$$

To make A ω-c.a.-tracing, meet requirements $P_{e,x}$ as Proposition 3.3

$$P_{e,x}\colon\ f_e(x) \in T^A_x\ (e \leq x).$$

We also use the same P strategies as Theorem 3.3 for P_e but on a tree. Each of the nodes will guess if Φ_e is total or partial.

To aid us in meeting the S requirements, we introduce a few new definitions. Following standard conventions, in particular assuming that all uses at stage s are bounded by s. Define the agreement function

$$l(e,s) = \max\{n : \forall k < n(\Phi^A_e(k)[s]\downarrow)\}$$

a maximum length of agreement function

$$m(e,s) = \max\{l(e,t) : t \leq s\}$$

We say that a stage s is *e-expansionary* if $l(e,s) > m(e,s-1)$. That is $\Phi^{A_s}_{e,s}$ has converged for a longer initial segment n of $\mathbb{N}$ than all previous stages.

The key idea for meeting S_e is as quite simple: wait until an e-expansionary stage s, allow S_e to act by tracing each newly converged $\Phi^A_e(n)[s]$ values into the appropriate trace set V_n, then initialize all weaker requirements.

Notice that we can not guarantee that the length of agreement associated with a given requirement is finite. It may well be the case that $\lim_s l(e,s) = \infty$, and S_e initializes a weaker requirement infinitely often. Therefore not giving a weaker P requirement a chance to act.

To aid weaker P requirements to deal with this, we use a tree of strategies. Consider a $P_{i,x}$ below S_e, we have two strategies for $P_{i,x}$, one guess that Φ_e is partial, i.e. there are only finitely many e-expansionary stages. This strategy puts $f_i(x)$ into T^A_x with large use each time $f_i(x)$ changes its mind and attempts to remove the previous value. Since it believes that Φ_e is partial, S_e will not attempt to trace anything and will not initialize any requirements, therefore $P_{i,x}$ can freely remove its previous value by a A change.

The strategy guessing total is a little trickier to deal with. The idea is we ask a strategy $\beta\colon P_{i,x}$ below the infinitary outcome of a strategy $\alpha\colon S_e$ to wait. Give Φ^A_e enough time to converge for large enough y, then trace $z = f_e(x)$ values, if f_e changes, remove z from the trace with an A-change. This allows f_e to change up to y times. Notice that each time it changes, it will only effect the values of $\Phi^A_e(x)$ for $x \geq y$. Therefore these A changes won't make $|V_y|$ too large.

Since f_e is a ω-c.a. function, we have in advance a computable function h_e such that $f_e(x)$ can change its mind at most $h_e(x)$ times. If we were dealing with just one β, we would naturally let $y = h_e(x)$ as if it were just a single $\beta \succ \alpha^\frown\infty$, for each i $\beta : P_{e,x}$ can not change $\Phi^A_i \restriction_y$ where y is the computable function $h_e(x)$ bounding the number of mind changes of $f_e(x)$. So we get at most y enumerations into V_y.

Notice that there are multiple β below α that could potentially make an A-change. So with an eye to the more complicated case, we set our y to not only consider the mind changes that $f_e(x)$ can make but also specifying the strategy β and the number of stages it was initialized, as each initialization may cause an A-changes. We denote by $k_{init}(\beta)$ the number of stages at which β has been initialized. We let y be the encoding of the triplet $\langle h_e(x),\beta,\mathrm{k}_{init}(\beta)\rangle$

We now turn to the formal details of the construction.

The Priority Tree. We use the tree $T = \{\infty, f\}^{<\omega}$. ∞,f denote the total and finite outcomes respectively. To each node $\sigma \in T$, we assign strategies N_σ and $P_{\sigma,x}$ for $N_{|\sigma|}$ and $P_{|\sigma|,x}$ respectively. Using a lexicographic ordering, with ∞ to the left of f.

We also define the notion of σ-stage, $m(\sigma,s)$ and *σ-expansionary stage* by induction on $|\sigma|$.

i) Every stage is a null-stage.

ii) Suppose that s is a τ-stage. Let e=$|\tau|$. Let

$$m(\tau, s)= \max\{l(e,t) : t < s \text{ is a } \tau\text{-stage}\}$$

If $l(e,s) > m(\tau,s)$ then we declare s to be τ-expansionary and declare s to be a $\tau^\frown\infty$-stage. Otherwise, we declare s to be a $\tau^\frown f$-stage.

Let the true path at stage s, TP_s, be the unique σ of length s such that s is a σ-stage.

Definition 3.12. Let the true path TP be the leftmost path of T visited infinitely often. I.e. let TP be the unique path of T such that

$$\sigma \prec \mathrm{TP} \text{ iff } \exists^\infty s(\sigma \prec \mathrm{TP}_s) \wedge \exists^\infty s(\mathrm{TP}_s <_{\mathrm{lex}} \sigma), \text{ and } s \text{ is a } \sigma \text{ stage}$$

where $<_{\mathrm{lex}}$ is the lexicographical ordering.

Definition 3.13. We say that $P_{\sigma,x}$ *requires attention* at a σ-stage $s > |\sigma|$ if $f_{|\sigma|}(x)[s-1] \neq f_{|\sigma|}(x)[s] \downarrow= y$.

Construction. At stage s, proceed as follows.

Step 1. Compute the current true path TP_s, then initialize all strategies nodes attached to nodes to the right of TP_s.

To initialize a $\alpha : P_{e,x}$ node, stop the computation of $f_e(x)$, remove the current value (if any) from $T_z^{A_s}$ and reset $f_e(x) = 0$.

To initialize a $\beta : N_e$ node, halt on tracing $\Phi_e^{A_s}$. The next time it is eligible to act, at say stage k, restart tracing $\Phi_e^{A_k}$ from 0.

Step 2. Check if any nodes is eligible to act. A strategy τ of length t be eligible to act at substage t of stage $s \geq t$ if τ has the correct guess about the current outcomes of all $\sigma \subset \tau$. in addition:

A $\alpha : P_{e,x}$ strategy node is eligible to act if it either believes that Φ_e^A is partial or $\Phi_e^{A_s}$ has converged for large enough y. That is, $\Phi_e^{A_s}$ has converged for each $x \leq y$, where y is an encoding of the triplet $\langle h_e(x), \alpha, k_{\mathrm{int}}(\alpha)\rangle$

A $\beta : N_e$ strategy is eligible to act if it is an e-expansionary stage.

If no strategy is eligible to act, continue to stage $s+1$.

Step 3. For each strategy that is eligible to act, they act depending on the type of strategy:

A $\alpha : P_{e,x}$ strategy will begin computing $f_e(x)$. Whenever it requires attention, remove its previously converged value with an A-change and enumerate $y \in T_x^A$ with high use on A. In other words, $\alpha : P_{e,x}$ acts each time it requires attention. Once $\alpha : P_{e,x}$ has acted, initialize all nodes properly extending α

A $\beta : N_e$ strategy will trace each new value $\Phi_e^{A_s}(x)$ that converged during the e-expansionary stage into V_x.

End of Construction.

Verification. We first show that all values has been traced and that the traces are bounded.

Claim 3.14. *Each P-requirement has been met.*

Proof. For an arbitrary $\alpha : P_{e,x}$ node, we assume that Φ_e^A is total. Choose a stage s such that $\Phi_e^{A_s}$ has converged for large enough y and TP_t is not to the left of α for all $t > s$. We can assume that we are on the true path and it has had the correct guess about Φ_e^A up until now. Hence such a stage must exist.

Now, since both requirements for eligibility have been met, $\alpha : P_{e,x}$ is now eligible to act. By the minimality of s, this is the first time $\alpha : P_{e,x}$ has been visited. So it will begin computing $f_e(x)$. As f_e is an ω-c.a. function, it has only finitely many mind changes.

Consider the true path in the limit, the function $f_e(x)$ will make its final mind change, yield a value, and $\alpha : P_{e,x}$ will require attention. The value will be traced into T_x^A and the previous value removed. Thus satisfying $\alpha : P_{e,x}$.

Claim 3.15. *Each* $|T_x^A| \leq x \times 2^x$.

To calculate the bound we count the number of nodes capable of contributing elements.

There are at most x functions f enumerating elements into T_x^A as we are only tracing $f_e(x)$ for $e \leq x$. A P strategy node not on the eventual true path may have started its computation, enumerated an element into T_x^A, and was never visited again. There are at most 2^x number of these nodes on a binary tree. Since at most x functions needs to be traced, we can conclude that:

$$|T_x^A| \leq x \times 2^x$$

□

Claim 3.16. *Each N-requirement is met, each* $|V_y| \leq y^2 \times 2^y$.

Proof. Consider N_e. Let $\beta = \mathrm{TP}\restriction_e$. If $\beta^\frown 0 \prec \mathrm{TP}$ then there are only finitely many e-expansionary stages, so Φ_e^A is not total, it need not be traced and N_e is met.

The only interesting case is when $\beta^\frown\infty \prec \mathrm{TP}$ and Φ_e^A is total. We show that each $\Phi_e^A(y)$ has been traced with a computable bound. Let s be the least stage such that no P-strategy node attached to a prefix of β acts after stage s and TP_t is not to the left of σ for all $t > s$. This stage exists as each positive strategy node acts finitely often.

At this stage, if $\Phi_e^{A_s}(y)$ has been traced then we are done, as no P-strategy nodes will initialize the requirement. If not, we wait until the next e-expansionary stage s' such that $\Phi_e^{A_{s'}}(y)\downarrow$, this must exist as Φ_e^A is total, at this stage $\beta : N_e$ becomes eligible to act and acts by tracing $\Phi_e^{A_{s'}}(x)$. Since β will no longer get initialized, we have $\Phi_e^A(y) = \Phi_e^{A_{s'}}(y)$. Thus $\Phi_e^A(y) \in V_y$ for each y and N_e is met.

For the bound on V_y we count the maximum number of initializations on the negative requirements.

By the construction, $\beta : P_{e,x}$ can not change $\Phi_i^A\restriction_y$ where y is encoding of the triplet $\langle h_e(x),\beta,\ \mathrm{k}_{init}(\beta)\rangle$. β is allowed to make a contribution when it makes a mind change. For each initialization there is at most $h_e(x)$ mind changes. Both $\mathrm{k}_{init}(\beta)$ and $h_e(x)$ is at most y, the contribution can be no greater than 2^y. Since we are using a binary tree, there are at most 2^y such β. Hence we have the desired bound

$$|V_y| \leq y^2 \times 2^y.$$

□

As shown above, the requirements have all been met, and both $|T^A_x|$ and $|V_y|$ are computably bounded, which concludes the proof of the theorem. □

3.2. **Non-ω-c.a.-tracing can be close to $\emptyset'$.**

In [21, 8.4.27], Nies showed that for any Δ^0_2 set A, superhighness is equivalent to jump traceable hardness, which implies ω-c.a.-tracing. Thus we can conclude that every superhigh Δ^0_2 set is ω-c.a.-tracing. This, however, does not translate to high c.e. sets. The following is in some way the dual of Theorem 3.1.

Theorem 3.17. *Some high c.e. set A is not ω-c.a.-tracing.*

Proof. This is similar to tree construction for building a high minimal pair. Define an ω-c.a. function g. We meet the requirements

$$N_e \colon \exists x\, g(x) \notin T^A_{e,x},$$

where $(T^A_{e,x})_{e,x\in\mathbb{N}}$ is uniform listing of all oracle c.e. traces with bound x.

We also meet the usual highness requirements

$$P_r \colon \mathrm{Inf}(r) = \lim_s \Gamma(A; r, s),$$

where $\mathrm{Inf} = \{r : |W_r| \text{ infinite}\}$ is the canonical Π_2-complete set.

Use methods from the tree construction of a high minimal pair as in [26]. Guess on the tree whether $\emptyset''(r) = 0$. This yields a notion of α-correct A-computations, where α is a string.

Strategy for α: P_r. Enumerate the r.e. set W_r and keep defining $\Gamma(A; r, s)$ to equal 0 for larger and larger s, with some use $u(r, s)$ bigger than any number mentioned thus far in the construction. Then

Step 1. Define a parameter n.

Step 2. Wait for a new number $\geq n$ to appear in W_r at some stage s, then for each $s' \leq s$, we enumerate the current use $u(r, s')$ into A (if currently the function $\Gamma(A; r, s') \downarrow= 0$) and redefine $\Gamma(A; r, s') = 1$ with the same use as before.

Step 3. Increment n by $+1$ and go back to Step 2.

Notice that the axiom defining $\Gamma(A; r, s')$ does not depend on A. As long as the strategy is prevented from redefining $\Gamma(A; r, s')$ from 0 to 1 at most finitely often, it will ensure the requirement.

We say that the outcome of a P_r-strategy is *finitary*, denoted as the 1 strategy node, if the strategy resets $\Gamma(A; e, s') = 1$ only for $s' \leq$ some fixed v, and the parameter n reaches a finite limit n_0. Note that this means $n_0 - 1 = \max(W_r)$ and W_r is finite. Similarly, we say that the outcome of a P_r strategy is *infinitary*, denoted as the 0 strategy node, if the strategy resets $\Gamma(A; e, s') = 1$ for all s'. Note that this will mean the set W_r contains arbitrarily large numbers and is thus infinite.

In either case, we have $\text{Inf}(r) = \lim_s \Gamma(A; r, s)$ as required.

We let the *current outcome* of the P-strategy at stage s be finitary if the parameter n remains unchanged at n_0 during stage s. Otherwise, we say it is the infinitary outcome, denoted ∞. So a true finitary outcome of a strategy is the current outcome of the strategy at co-finitely many stages whereas a true infinitary outcome of the strategy is the current outcome only at infinitely many stages.

Strategy for α: N_e. Pick large x. Define $g(x)$ to be a fresh large number y. Whenever $g(x) = y \in T^A_{e,x}$ via an α-correct computation, we believe that y can not leave $T^A_{e,x}$. Hence we can increase $g(x)$ and *initialize* all weaker requirements as follows:

To initialize a $\beta : N_e$ strategy, force g to choose a fresh large number x, define $g(x) = 0$.

We denote s_α at stage s to be the least stage $s' \leq s$ such that α is on the true paths at stage s' and α has not been initialized since stage s.

Construction. Our *tree of strategies* T will be a subtree of the binary tree $2^{<\omega}$, where 0 and 1 will denote the infinite and finite outcomes respectively. At stage s, we define a stage $s_\sigma = s_\sigma[s]$ to be the least stage $s' \leq s$ such that $\sigma \subseteq \delta_{s'}$ and σ hasn't been initialized since stage s'.

Stage $s = 0$. For each r, let the parameter $n = 0$ for all P_r, $\Gamma(A, r, 0) = 0$ with use 0 and $g(x) = 0$ for all x.

Stage $s+1$. First define the current true path $\delta_s \in T$. Let a strategy σ of length t be *eligible to act* at a sub-stage t of stage $s \geq t$ iff σ has the correct guess about the current outcomes of all $\tau \subset \sigma$. In addition:

A $\beta : N_e$ strategy is eligible to act if an α-correct computation $g(x) = y \in T^A_{e,x}$.

Search for the strategy of the least length that is eligible to act. Then perform actions depending on the strategy type:

An $\alpha : P_r$ strategy *acts* by performing step 1 to 3 as described in the strategy as follows:

Let s_0 be the greatest stage $t \leq s$ such that $\alpha \subseteq \delta_{s_0}$, if no such stage exists, then move on to stage $s+1$. Suppose α was first eligible to act at stage s, we set the parameter n to be the last stage it was initialized, denoted $s_{init}(\alpha)$. If no elements enters W_{x,s_0}, we set $\Gamma(A; x, s') = 0$ with the previous use (for all $s' \leq s$ such that $\Gamma(A; x, s')$ is currently undefined) unless $\Gamma(A; x, s') = 0$ has not been defined before stage s, in which case we select a fresh large number $u(r, s')$ as its use and let $\alpha^\frown\langle 1\rangle$ be *eligible to act* next.

Otherwise, if some n enters W_{x,s_0}, enumerate the use $u(x, s')$ into A for all $s' \geq s_\alpha$ such that $\Gamma(A; x, s') = 0$. Then define $\Gamma(A; , x, s) = 1$ with use -1 for all $\Gamma(A; x, s)$ that is now undefined and let $\alpha^\frown\langle 0\rangle$ be *eligible to act* next.

In addition, as a protection procedure, at the end of any stage, we force the strategy to redefine $\Gamma(A;r,s)$ to its previous value with the same use if it is now undefined due to an A-change.

A $\beta : N_e$ strategy *acts* by redefining $g(x)$, since the computation for $g(x) = y \in T^A_{e,x}$ is α -correct, the strategy believes in the computation. Select a fresh large number y', set $g(x) = y'$ and *initialize* all weaker requirements.
End of Construction.

Verification. Define the *true path* $f \in [T]$ of the construction by:
$f(n) = \mu k \leq 1$ such that $f\restriction_n{}^\frown\langle k\rangle, k \in \{0,1\}$, is eligible to act infinitely often.

Lemma 3.18. ***(True Path Lemma).***
(i)The true path f is well-defined.
(ii) For all strategies $\sigma \subset f$, it is initialized finitely often and thus $s_\sigma[s]$ eventually reaches a finite limit.

Proof. (i) Our tree T is finitely branching, and $\lim_s|\delta_s| = \infty$.
(ii) Apply induction procedure on $|\sigma|$. Assuming a stage where no $\tau \subset \sigma$ is initialized by any stronger strategies. σ can only be initialized in a future stage s only if $\delta_s < \sigma$. This can only happen at finitely many stages by the definition of f. □

Lemma 3.19. *For each r, Inf(r) = $lim_s\Gamma(A;r,s)$, and hence A is high.*

Proof. Fix r and the P_r strategy $\alpha \subset f$,
First assume that W_r is finite. Choose a stage $s_0 \geq s_\alpha$ such that $\alpha \subset \delta_{s_0}$ and $W_r = W_{r,s_0}$. At this point, no more $\Gamma(A;r,s)$ will be defined to $= 1$ by any P_r strategy.
Now assume that W_r is finite. By step 2 of the construction, $\Gamma(A;r,s) = 1$ for all $s \geq s_\alpha$ with use -1. □

Claim 3.20. *The function g is ω-c.a.*

First notice that as long as α: N_e is not initialized, all relevant computations $y \in T^A_{e,x}$ are α-correct. The size of $T^A_{e,x}$ is bounded by the order function 2^x. So the α-correct computations will increase $g(x)$ at most 2^x times.

However, we need to take into account the interaction between strategies. Consider a $\beta : N_e$ strategy below the 0 outcome of an $\alpha : P_r$ strategy. Let s be the least stage where β becomes *eligible to act* for the first time or when $g(x) = y$ enters $T^A_{e,x}$.

Since each α strategy is $s_{init}(\alpha)[s']$ at some stage s', and $s_{init}(\alpha) \leq s$. The use that P_r strategies puts into A are numbers too small to affect the membership of any elements in $T^A_{e,x}$ that interacts with $g(x)$.

Also, β strategies always force $g(x)$ to redefines itself large, at stage s, β chooses a new value for $g(x)$ larger than all numbers seen in the computation so far, which by definition is larger than any already defined use value for all α strategies above.

Hence, by the argument above, the relevant A computations are protected, if a new value $g(x)$ were to enter $T^A_{e,x}$, it can not leave allowing us to redefine

$g(x)$ and not worry about the bound on the number of changes as it will remain $\leq |T^A_{e,x}|$. $\square$

An alternative way to the proof was suggested by Nies in the 2012 Logic Blog, instead of using a tree of strategies, one could set:

$$P_r : \emptyset''(r) = \lim_n A^{[r]}(n),$$

and use an α-correct approach on $g(x)$. However, the above method has the advantage of being more intuitive, and hence was chosen to be presented in this thesis.

4. Schnorr randomness relative to $\emptyset'$

The main focus of this section of the thesis is Schnorr randomness. It is a notion proposed by German mathematician Claus P. Schnorr. He first introduced it as an alternative approach to the definition of an algorithmically random sequence. Recall that a Schnorr test is the same as a Martin-Löf test with $\lambda(U_n) = 2^{-n}$ with the same pass condition. We also define the concept of full relativization for a randomness notion $\mathcal{C}$ to an oracle A. This is denoted by $\mathcal{C}[A]$ and indicated with the phrase "relative to A".

Here we will show that with a full relativization approach, we can find some equivalences between Martin-löf randomness, Schnorr randomness and limit randomness. Limit randomness s a notion derived from Demuth randomness, similar to the underlying idea in the limit lemma, we decide that the tests should not be fixed, but allowed to change some amount of times:

Definition 4.1. A limit test is a uniform sequence $\{U_n\}_{n\in\mathbb{N}}$ of c.e. classes such that for each $n \in \mathbb{N}$, $\lambda(U_n) \leq 2^{-n}$, and there is a function $f \leq_T \emptyset'$ such that $U_n = [W_{h(m)}]^{\prec}$.

A set A passes the limit test if it passes in the Solovay sense: $A \notin U_n$ for all but finitely many n.

We shall denote the class of sets that is Schnorr random relative to $\emptyset'$ as SR[$\emptyset'$].

The main goal here is to demonstrate that for all low A, MLR^A, $SR[\emptyset']$ and limit random are the same notions. As mentioned in the introduction, the first result here will be quite useful later in section 6.

Theorem 4.2. *A is limit random if and only if A is Schnorr random relative to $\emptyset'$.*

Proof. ($\Leftarrow$) :Given a limit test $\{U_n\}_{n\in\mathbb{N}}$, for each U_n, we can find a $h \leq_T \emptyset'$ such that $W_{h(n)} = U_n$. Now, construct a SR[$\emptyset'$] test as follows:

$$\text{Let } R_m = \textstyle\bigcup_{k>m} W_{h(k)} \ .$$

Since $h \leq_T \emptyset'$, this is a $\Sigma^0_1(\emptyset')$ set. So all that is left is to ensure that λR_m is computable in $\emptyset'$. In other words, we need to find a sequence of $\emptyset'$ computable rationals $q_0, q_1, q_2, q_3, ... \to R_m$ such that $\mid R_m - q_n \mid < 2^{-n}$ for all n.

Notice that for $\lambda(\bigcup_{m<k<n} W_{h(k)})$:

$$\lambda R_m - \lambda(\textstyle\bigcup_{m<k<n} W_{h(k)}) = \lambda(R_m - \bigcup_{m<k<n} W_{h(k)}) \leq \lambda(\bigcup_{p>n} W_{h(p)})$$

Since $W_{h(m)}$ is a limit test $\lambda(\bigcup_{p>n} W_{h(p)}) \leq 2^{-n}$.

However, $\lambda(\bigcup_{m<k<n} W_{h(k)})$ may not be a rational, but we will approximate it by letting $q_n = \lambda(\bigcup_{m<k<n} W_{h(k)}) \restriction n$, that is, q_n is given by the first n bits of $\lambda(\bigcup_{m<k<n} W_{h(k)})$. $\lambda(\bigcup_{m<k<n} W_{h(k)} - q_n \leq 2^{-n})$. So $\lambda R_m - q_n \leq 2^{-n} + 2^{-n} = 2^{-n+1}$.

Hence $(R_m)_{m\in\mathbb{N}}$ is a SR[$\emptyset'$] test.

For the other direction we use the Schnorr version of the Solovay test and prove the contrapositive;

($\Rightarrow$): Let A be not SR[$\emptyset'$] random, so it does not pass all $\emptyset'$ total Solovay tests. Therefore there is a $\emptyset'$-computable c.e. open sets $(\sigma_i)_{(i \in \mathbb{N})}$such that the sum $\sum_{i=0}^{\infty} \lambda(\sigma_i)$ is finite, a computable real and $\sum 2^{-|\sigma_i|}$ is computable in $\emptyset'$. For A to fail the test it has to extend infinitely many σ_i.

Hence, we will use $\emptyset'$ to find a sequence $n_0 < n_1 < n_2 ....$ such that

$$\textstyle\sum_{i=n_k}^{n_k+1} 2^{-|\sigma_i|} \leq 2^{-k}.$$

We let:

$$G_k = [\{\sigma_i : n_k \leq i < n_{k+1}\}]^{\prec}$$

So for each k, $\lambda(G_k) \leq 2^{-k}$ and $(G_k)_{k \in \mathbb{N}}$ is a limit test. Since A extends infinitely many σ_i, there exist a k such that $A \in G_k$. Hence, A is not limit random. □

Further in Yu Liang's paper [31, Thm 4.1], he showed that the equivalence does not stop there. In fact, if a set A is low, then we can construct a MLR^A test that covers all SR[$\emptyset'$] tests.

Theorem 4.3. *$Z \in SR[\emptyset']$ iff $Z \in \mathsf{MLR}^A$ for all low A.*

The following is a modified version of the finite injury argument used by Yu in the original construction. We do not change the main approach but add more details to the construction and verification.

Proof. We first start with the easier direction:

($\Rightarrow$:) A is low, $A' \leq \emptyset'$, so each $\lambda(V_n^A)$ is computable from $\emptyset'$.

Suppose we have a low A and a MLR^A test $\{V_n^A\}_{n \in \mathbb{N}}$, we build a SR[$\emptyset'$] $\{U_{n'}^A\}_{n' \in \mathbb{N}}$ test as follows:

For each V_n^A ask if "$\lambda(V_n^A) > 2^{n'}$" for each $n' \in \mathbb{N}$, this process is Σ_1^0. Next, assign $V_n^A \subseteq U_{n'}^A$ where $\lambda(V_n^A) > 2^{n'}$ and n' is maximal.

Lastly, enumerate arbitrary elements into $U_{n'}^A$ to ensure that $\lambda(U_{n'}^A) = 2^{n'}$.

We can now see that $(U_{n'})_{n' \in \mathbb{N}}$ is a $\emptyset'$-computable Σ_1^0 sequence with each $\lambda(U_{n'}^A) = 2^{n'}$. Hence, a SR[$\emptyset'$] test.

Each $V_n^A \subseteq U_{n'}^A$ for some n', thus the proof is complete.

The reverse direction is proved using a finite injury argument:

($\Leftarrow$:)For every $\emptyset'$-Schnorr test $\{U_n^{\emptyset'}\}_{n \in \mathbb{N}}$, we construct a low A and a MLR^A test $\{V_n^A\}_{n \in \mathbb{N}}$ such that $\bigcap_{n \in \mathbb{N}} U_n^{\emptyset'} \subseteq \bigcap_{n \in \mathbb{N}} V_n^A$.

Without loss of generality, we force each $\lambda(U_n^{\emptyset'})$ to equal 2^{-2^n} and $U_{n+1}^{\emptyset'} \subseteq U_n^{\emptyset'}$. The following requirement is sufficient:

$$P_e \colon U_e^{\emptyset'} \subseteq V_e^A;$$

We also meet the usual lowness requirement as in Theorem [3.3]:

$$N_e \colon (\exists_s^{\infty} \Phi_e^A(e)[s] \downarrow) \Rightarrow \Phi_e^A(e) \downarrow.$$

The idea is that we first decompose P_e into infinitely many sub-requirements $P_{\langle e,n\rangle}$. At each $P_{\langle e,n\rangle}$, define the subset

$$U_e^{\emptyset'} \restriction n = \{\sigma \in 2^\omega : |\sigma| \leq l_{e,n} \wedge \sigma \in U_e^{\emptyset'}\}$$

where $l_{e,n}$ is the least l such that $\lambda(U_e^{\emptyset'} \cap 2^{\leq l}) > 2^{-e}(1-2^{-2^n})$. This measure converges to the true measure of $U_e^{\emptyset'}$ as $n \to \infty$. At stage s, we can now define the relative $\emptyset'$-Schnorr test:

$$U_e^{\emptyset'}[s] \restriction n = \{\sigma \in 2^\omega : |\sigma| \leq l_{e,n}[s] \wedge \sigma \in U_e^{\emptyset'}[s]\}.$$

Therefore each sub-requirement is of the form:

$$P_{\langle e,n\rangle} : U_e^{\emptyset'} \restriction n \subseteq V_e^A$$

Notice that, in the limit, it's sufficient to just satisfy those $P_{\langle e,n\rangle}$'s such that $e \geq n$.

With the above set up, we now construct a function f to aid the positive requirements as follows:

$$f(e,\sigma,s,n) = \begin{cases} 1 & \text{if } \sigma \in U_e^{\emptyset'}[s] \restriction n \\ 0 & \text{otherwise} \end{cases}$$

The function serves as a tracker to help to build the set V_e^A accordingly whenever a σ enters or leaves $U_e^{\emptyset'}$. Next, we move on to setting up the negative requirements. As per usual in finite injury arguments, we set up the restraint function $r(e,s) > \varphi_e^{Z_s}(e)$ where $\varphi_e^{Z_s}(e)$ is the use function of $\Phi_e^Z(e)[s]$. Now define:

$$R(e,s) = \textstyle\sum_{i \leq e} r(i,s).$$

This will serve as a restraint to protect the computation $\Phi_e(e)[s]$.

Strategy for N_e: Use the same strategy as Theorem 3.3 except with the restraint function $R(e,s)$.

Strategy for P_e: We use a similar method used in the Friedberg and Muchnik solution to Post's Problem. Define our function with input of the form $\langle e,\sigma,t_s\rangle$. At any stage s, instead of attaching a single σ as a follower, we attach a triplet $\langle e,\sigma,t_s\rangle$ to σ. The t_s acts as a tracker for whenever we need to attribute a new follower. Whenever $f(e,\sigma,s,n)=1$ and $f(e,\sigma,s,n')=0$ for all $n' < n$, set $z_s(\langle e,\sigma,t_s\rangle)=1$ and $z_s(\langle e,\sigma,t_s\rangle)=0$ if σ exists. Our Martin-Löf test would then be defined as the set of triplets that yields 1 in A.

Construction At stage s, proceed as follows: For any σ with $l_{e,n}[s] \geq |\sigma| > l_{e,n-1}[s]$, if either it has no follower at stage s or the follower has been initialized in stage $s-1$, we allocate a new follower $\langle e,\sigma,t_s\rangle$ to σ with fresh large number t_s; otherwise, let the old follower stay unchanged by setting $t_s = t_{s-1}$.

Next, find the requirement with the highest priority requiring attention. A $P_{\langle e,n\rangle}$-type strategy *requires attention* if one of the following holds:

1) σ enters $U_e^{\emptyset'}[s] \restriction n - U_e^{\emptyset'}[s] \restriction (n-1)$ but $z_s(\langle e,\sigma,t_s\rangle)=0$.
The action is to set $z_{s+1}(\langle e,\sigma,t_s\rangle)=1$; or
2) σ leaves $U_e^{\emptyset'}[s] \restriction n$ but $z_s(\langle e,\sigma,t_s\rangle)=1$.
The action is to set $z_{s+1}(\langle e,\sigma,t_s\rangle)=0$.

We say that $P_{\langle e,n\rangle}$ has *received attention*. To avoid conflict between $P_{\langle e,n\rangle}$ and a weaker $P_{\langle e',n'\rangle}$, once $P_{\langle e,n\rangle}$ has received attention, initialize all the parameters for $P_{\langle e',n'\rangle}$.

A N_e-type strategy *requires attention* at stage s if $\Phi_e^A(e)[s]\downarrow$ but N_e has not received attention after it has been initialized before stage s. Restrain all weaker P-type strategies from changing A by initializing all weaker strategies. This way, since we assume that the use of the computation at stage s can not exceed s, all relevant computations of A_s are protected. We then say that N_e *receives attention.*

To *initialize* a $P_{e,x}$-type strategy, reset its follower by setting $z_{s+1}(\langle e,\sigma,t_s\rangle) = 0$. Then, select a fresh large number k, let $t_{s+1} = k$.

To *initialize* a N_e-type strategy, cancel its computations on $\Phi_e^A(e)$ (this includes resetting the using function) and restart the computation during stage $s+1$ with $R(e,s) = 0$.

If no requirement needs attention, do nothing, proceed with next stage with $z_s = z_{s+1}$ and $t_s = t_{s+1}$.

Finally, define: $V_e^A = \{\sigma \mid \exists s[z_s(\langle e,\sigma,t_s\rangle = 1)]\}$.

End of Construction.

Verification. We first show that the injury set is finite, then show each requirement has indeed been met. Define the injury set I_e for N_e in the same way as in Theorem 3.3.

Claim 4.4. *The injury set I_e is finite for all e.*

Proof. We argue by looking at the injuries caused by each $P_{\langle e,n\rangle}$. Suppose $P_{\langle e,n\rangle}$ is a stronger requirement than N_e. $P_{\langle e,n\rangle}$ injures N_e each time $\langle e,\sigma,t_s\rangle$ changes membership in $U_e^{\emptyset'} \restriction n$. Since $U_e^{\emptyset'}$, is a $\emptyset'$-Schnorr test, this happens finitely often. As there are only finitely stronger $P_{\langle e,n\rangle}$ requirements, the injury set must be finite.

Also, notice that if $P_{\langle e,n\rangle}$ ever becomes initialized at some stage s, it would choose a fresh large number t_{s+1}, this number will ensure it can not injure any computation in N_e up to stage s as they are less than t_{s+1}. □

Claim 4.5. *For every e, requirement N_e is met and $r(e) = lim_s r(e,s)$ exists.*

Proof. The argument is the same as in Theorem 3.3: fix e, by our first claim, choose a stage s_e such that N_e is not injured after stage s_e. If $\Phi_{e,s}^{A_s}(e)$ converges for $s > s_e$, then by induction on $t \geq s$, $r(e,s) = r(e,t)$ and $\Phi_{e,t}^{A_t}(e) = \Phi_{e,s}^{A_s}(e)$ for all $t \geq s$. So $A \restriction r = A_s \restriction r$. Hence $\Phi_e^A(e)$ is defined by the *use principle.* □

Claim 4.6. *Each positive requirement is satisfied and $\{V_e^A\}_{e\in\mathbb{N}}$ is a A-Martin-löf test.*

Proof. $U_e^{\emptyset'} \subseteq V_e^A$ for every e by definition. So all that is left to check is that $\{V_e^A\}_{e\in\mathbb{N}}$ has the appropriate measure.

First not that each $P_{\langle e,n\rangle}$ contributes no more than $2^{-2^e-(2^n-1)}$ measure of clopen sets into V_e^A for any pair $\langle e,n\rangle$.

Secondly, by Claim 4.4, each N_e is injured finitely often. For each $P_{\langle e,n\rangle}$ with $e \geq n$, there are e many negative requirements with higher priority. Once a stronger requirement set up a restraint function, $P_{\langle e,n\rangle}$ can not change its parameters less than the restraint. So $P_{\langle e,n\rangle}$ can make at most 2^n mistakes. Thus for $e > 1$ we have:

$$\lambda(V_e^A) \leq \sum_{n\in\mathbb{N}}(2^n) \times 2^{-2^e-(2^n-1)} \leq \sum_{n\in\mathbb{N}} 2^{-2^e-n+1} = 2^{-2^e+2} \leq 2^{-e}.$$

□

We have shown that $\{V_e^A\}_{e\in\mathbb{N}}$ is a ML-test for $e \geq 2$, the requirements are satisfied and $\bigcap_{n\in\mathbb{N}} U_n^{\emptyset'} \subseteq \bigcap_{n\in\mathbb{N}} V_n^A$. This, with the previous direction gives us the theorem.

□

5. Randomness and cuppablity

We begin this section with the necessary definitions:

Definition 5.1. We say A is (weak) Demuth cuppable if there is a (weak) Demuth random set Z such that $A \oplus Z \geq_T \emptyset'$. If one can choose $Z <_T \emptyset'$, then A is Demuth cuppable.

Definition 5.2. A set A is weakly ML-cuppable if there is an incomplete Martin-Löf random set Z such that $A \oplus Z \geq_T \emptyset'$. If one can choose $Z <_T \emptyset'$, then A is ML-cuppable.

In this section, we study in to the relationship between Demuth random sets, superlow sets and the K-trivials. We work towards the following result, that for any c.e. set:

$$K\text{-trivial} \Rightarrow \text{weak Demuth noncuppable} \Rightarrow \text{superlow},$$

Recall that A is K-trivial if $K(A \upharpoonright n) \leq K(n) + c$ for some constant c. Where K is the prefix-free Kolmogorov complexity. For the sake of this section, it is easier to think of the K-trivials as the sets that contain no random content. Kuĉera originally hypothesised that the notion of Demuth noncuppable can be used to characterize K-triviality. However the first formal proposition of the idea was posed as an open question in Nies and Miller's paper [19]. The question was then investigated by Adam Day and Joel miller, in their paper [6]. There, the first implication was proved, however, the result they arrived was stronger than originally expected. They showed that K-trivial even implies Martin Löf-noncuppablility:

Theorem 5.3. [6, Day/Miller, Theorem 3] *No K-trivial set is weakly ML-cuppable.*

The proof of the theorem builds on work of Franklin and Ng, and Bienvenu [14], Hölzl, Miller and Nies [5]. These are some of the tools that were used in the Day/Miller result:

Definition 5.4. We define the lower (Lebesgue) density of ρ of a effectively closed set $\mathcal{D}$ in the unit interval at a point x to be the quantity:

$$\rho(x \mid \mathcal{D}) := \liminf_{\delta \to 0^+} \frac{\lambda([x-\delta, x+\delta] \cap \mathcal{D})}{\lambda([x-\delta, x+\delta)}$$

This notion measures what fraction of the space is filled by $\mathcal{D}$ around x if we zoom in in the limit. Note that the density of a set at a point is always between 0 and 1.

We say that x is a point of positive lower density in $\mathcal{D}$ if $\rho(x \mid \mathcal{D}) \neq 0$.

Theorem 5.5. *(Bienvenu, Hölzl, Miller, Nies). Suppose x is Martin-Löf random, then x is Turing complete iff x has lower density zero inside some effectively closed class $\mathcal{D}$.*

Proof. The idea is to show that, given rational ϵ, we can construct an effectively closed class $\mathcal{D}_\epsilon$ such that $x \in \mathcal{D}_\epsilon$ and $\lambda(\mathcal{D}_\epsilon \mid x\upharpoonright_n) < \epsilon$ for some n. It then suffice to let $\mathcal{D} := \cap_\epsilon \mathcal{D}_\epsilon$ for an effective list of ϵ that tends to 0.

Fix $\epsilon > 0$, in this construction, to build the list of $\mathcal{D}_\epsilon$, we first build an auxiliary c.e set W. By the recursion theorem, since x is complete, we can assume to know in advance a Turing reduction such that $\Gamma^x = W$.

The idea of the construction is that we want to lower the density of $\mathcal{D}_\epsilon$ around x, to do this we need to remove reals from $\mathcal{D}_\epsilon$ without inadvertently removing x itself. Using the fact that W is controllable, we keep observing the results of the reduction until we see a certain type of behaviour (reduction outputs 0) on all oracles except a fraction of ϵ. Since x computes W it certainly cannot among the 1-ϵ fraction of oracles with the special behaviour. So we can safely remove them from $\mathcal{D}_\epsilon$.

Formal construction. Instead of a direct construction, we give an effective procedure $P(\sigma, k, \epsilon)$ that enumerates the complement of $\mathcal{D}_\epsilon$.

Step 1 Pick a fresh number $n = (\sigma, k)$

Step 2 Enumerate the set V of reals y that extends σ and satisfies $\Gamma^y = 0$. By the use principle, this will always happen on whole open cylinders of y's. Therefore we can represent V by a prefix free enumeration of finite strings. As long as the conditional measure of V above σ does not exceed $1-\epsilon$, put every enumerated string into U_k and for each of them, start the procedure $P(\tau, k + 1, \epsilon)$.

Step 3 At stage s, if some new τ is found such that $\Gamma^\tau(n) = 0$ but the conditional measure $U_k[s] \cup \{\tau\}$ above σ exceeds $1-\epsilon$, then enumerate n into W, enumerate all of $U_k[s] \cup \{\tau\}$ into the complement of $\mathcal{D}_\epsilon$, and terminate the whole tree of procedures.

Verification. We show that the procedure yields the desired result. *Claim 1* guarantees the safety of x while *Claim 2* ensures that we will not wait forever, since in that case the measure of $\mathcal{D}_\epsilon$ would forever remain equal to 1. The idea of the argument is that if we do not eventually observe the behaviour, there is a descending chain of Martin-Löf tests $\{U_k\}_{k\in\mathbb{N}}$ covering x. This of course contradicts that x is Martin-Löf random.

Claim 1. x remains in $\mathcal{D}_\epsilon$ at all times during the construction: when we put a string τ into the complement of $\mathcal{D}_\epsilon$ during a procedure in step 3, this τ has to satisfy $\Gamma^\tau(n) = 0$, and we precisely make sure at that point that $n \in W$. Since Γ^x=W, this shows that $\tau \not\preceq x$.

Claim 2. The conditions for step 3 will eventually be met: (note that this is required as if for some prefix σ of x and some k, the procedure $P(\sigma, k, \epsilon)$ gets executed and reaches step 3, then we are done since step 3 ensures that $\lambda(\mathcal{D}_\epsilon \mid \sigma) < \epsilon$).

If $P(\sigma, k, \epsilon)$ is a procedure that is executed but never reached step 3, then $n = n(\sigma, k)$ never gets enumerated into W, therefore $\Gamma^x(n) = W(n) = 0$. Hence by step 2 it must be the case that some prefix σ' of x is put into U_k and procedure $P(\sigma', k + 1, \epsilon)$ is executed.

Observe that $(U_k)_{k\in\mathbb{N}}$ is a Martin-Löf test, as each time a string σ is enumerated in U_k, some extensions are enumerated into U_{k+1} by step 2 during procedure $P(\sigma, k + 1, \epsilon)$. Each of the strings above σ weighs at most

$1-\epsilon$, therefore we have $\lambda(U_{k+1}) \leq (1-\epsilon)\lambda(U_k)$ for all k. By induction this shows that $\lambda(U_{k+1}) \leq (1-\epsilon)^{k+1}$ for all k.

We have shown that if step 3 is never reached in any of the executed procedures $P(\sigma, k, \epsilon)$ with $\sigma \preceq x$, the $x \in [U_k]$ for all k. However this is a contradiction as x is Martin-Löf random.

□

Relativization of Theorem 5.5 gives the following corollary:

Corollary 5.6. *Fix A. If X is a set Martin-Löf random relative to A and $A \oplus X \geq_T A'$, then there exists a $\Pi^0_1(A)$ class P such that $X \in P$ and X has lower density zero in P.*

We need one more result in order to prove Theorem 5.3.

Definition 5.7. (Nies 2002) Let A and B be Turing oracles. We say that A is *LR-reducible* to B, written $A \leq_{LR} B$, if

$$\forall X (X \text{ is } B\text{-random} \Rightarrow X \text{ is } A\text{-random}).$$

Lemma 5.8. *Assume $A \leq_{LR} B$. Let $f : \omega \to \omega$ be recursive. For all A-r.e sets I such that $\sum_{i \in I} 1/2^{f(i)} < \infty$, there exists a B-r.e set $J \supseteq I$ such that $\sum_{j \in J} 1/2^{f(j)} < \infty$.*

Proof. We use the following fact from analysis. Given $0 < a_i < 1$, $i = 0, 1, \ldots..$ we have

$$\textstyle\sum_{i=0}^{\infty} a_i < \infty \text{ if and only if } \prod_{i=0}^{\infty}(1-a_i) > 0.$$

Let A, B, f, I be as in the hypotheses of the lemma. Without loss of generality we assume that $f(i) \neq 0$ for all i. Let $g(i) = \sum_{k=0}^{i-1} f(k)$, define a set D_i of strings as follows:

$$D_i = \{X \in 2^\omega \mid \exists n (X(n) = 1 \text{ and } g(i) \leq n < g(i+1)\ \}$$

Note that the function $g(i)$ is strictly increasing so the D_is are mutually independent and each D_i is clopen as $D_i = [Y]^{\prec}$ for some finite set $Y \subseteq \{0,1\}^*$.

Now we represent a set P Π^0_1 in A by defining $P = \bigcap_{i \in I} D_i$, by the original hypothesis, we assumed that $\sum_{i \in I} 2^{-f(i)} < \infty$, hence $\lambda(P) = \prod_{i \in I} \lambda(D_i)$. Since each the weight of each D_i only depends on the depth of $f(i)$ on 2^ω we have $\lambda(D_i) = 1 - 1/2^{f(i)}$, and so $\lambda(P) = \prod_{i \in I}(1 - 1/2^{f(i)})$.

Choose $Q \subseteq P$ be Π^0_1 in B, choose strings with non-zero weight so $\lambda(Q) > 0$. Let $J = \{i \mid D_i \supseteq Q\}$, J is B-r.e. and each $i \in I \Rightarrow i \in J$. Its easy to see that $\bigcap_{j \in J} D_j \supseteq Q$, so we have $\prod_{j \in J}(1 - 1/2^{f(j)}) = \prod_{j \in J} \lambda(D_j)$. Since the D_is are measurable sets, $\prod_{j \in J} \lambda(D_j) = \lambda(\bigcap_{j \in J} D_j)$.

We defined $Q \subseteq P = \bigcap_{i \in I} D_i$ so we have $\lambda(\bigcap_{j \in J} D_j) \geq \lambda(Q)$ and we also chose $\lambda(Q)$ to be always positive, hence $\lambda(\bigcap_{j \in J} D_j) > 0$.

By our earlier fact, we can conclude that $\prod_{j \in J}(1 - 1/2^{f(j)}) > 0$ iff $\sum_{j \in J} 1/2^{f(j)} < \infty$. □

With Lemma 5.8 and the following definition, we can arrive at Corollary 5.10 by taking f to be the identity function and set B as the halting problem.

Definition 5.9. Let I be a set of finite strings. We call I *bounded* if:

$$\sum_{i\in I} 2^{-|i|} < \infty.$$

Corollary 5.10. *Let A be a K-trivial set and W_A be a bounded set of strings that is c.e in A, then there exists a bounded c.e. set of strings W such that $W_A \subseteq W$.*

Now we finally have the tools needed to prove Theorem 5.3.

Proof. Nies showed that all K-trivial sets are low, so we define a K-trivial set A and a Martin-Löf random set R such that $A \oplus R \geq_T \emptyset'$. Then we show any such set R must be complete. By Theorem 2.19, all K-trivial sets are low for Martin-Löf randomness. Hence R is Martin-Löf random relative to A. By Corollary 5.5, there exists a $\Pi^0_1(A)$ class P_A such that $R \in P$ and R has lower density zero in P_A.

Let W_A be an A-c.e. prefix-free set. Construct a set P_A by taking all strings that do not properly extend any $\sigma \in W_A$, i.e. $P_A = \{X \in 2^\omega : (\forall \sigma \in W_A)\sigma \not\prec X\}$. Since W_A is prefix-free, its easy to see that it has bounded weight. So by Corollary 5.10, we can find a bounded c.e set of strings W such that $W_A \subseteq W$.

R is Martin-Löf random, so it passes all Solovay tests. Since W has bounded weight, its must be a Solocay test. Hence there must be finitely many initial segments of R in W. We defined P_A to force R and W_A to be disjoint, so no initial segment of R is in W_A.

Now, begin by removing from W all initial segments of R. (Notice that W is still a bounded weight c.e. superset of W_A.) Then define a $P \subseteq P_A$ to be all strings that do not properly extend any $\sigma \in W$, i.e. $P = \{X \in 2^\omega : (\forall \sigma \in W)\sigma \not\prec X\}$. Since $R \in P$ and R has lower density zero in P_A, R has lower density zero in P. So by Theorem 5.5, R is a complete Martin-Löf random set.

As R is chosen arbitrarily, A K-trivial, we can conclude that any R such that $A \oplus R \geq_T \emptyset'$ is complete, hence A can not be weakly ML-cuppable. □

The second implication that weak Demuth noncuppable $\Rightarrow$ superlow is shown in [12]. They use that for c.e. sets A, superlow = ω-c.a. jump dominated.

Definition 5.11. A set A is ω-c.a.-jump dominated if there is an ω-c.a. function $f(x)$ and $\Phi^A_x(x) \leq f(x)$ for each x that $\Phi^A_x(x)\downarrow$.

Proposition 5.12.
(i) Every superlow set is ω-c.a.-jump dominated.
(ii) For c.e. sets, the converse implication holds as well.

Proof. (i) Suppose $\{A_s\}_{s\in\mathbb{N}}$ is a computable approximation of a superlow set A, and f is a computable function such that $\lim_s f(x,s) = A'(x)$ for every x, with computably bounded number of mind changes. Let φ_e be the e-th partial computable function. We define uniformly c.e sets $U_{i,x,e}$ as follows. For each s such that

1. $\varphi_e(x)[s]\downarrow$,
2. $f(\varphi_e(x), s)$=0, ($\varphi_e(x) \notin A'$ at stage s)
3. $|\{t < s : f(\varphi_e(x), t) \neq f(\varphi_e(x), t+1)\}| \leq 2i$, (the number of mind changes at stage s for f is bounded by $2i$) and

4. $J^{A_s}(x)[s] \downarrow$,

we enumerate the shortest initial segment σ of A_s such that $J^\sigma(x)[s] \downarrow$ into $U_{i,x,e}$. Now define $r(e,x)$ to halt and output x iff 1. 2. 3. holds. This way $\Phi_e^\sigma(r(x,e)) \downarrow$ iff some $\tau \preceq \sigma$ is in $\bigcup_i U_{i,x,e}$. By the recursion theorem, we will fix e such that $r(x,e){=}\varphi_e(x)$ for all x.

Define a function g as follows. If $r(x,e) \notin A'$ then let $g(x) = 0$. Otherwise, there is an i such that there are exactly $2i{+}1$ (i.e. odd) many $f(r(x,e),-)$ changes. Let $g(x)$ be the maximum of all $J^\sigma(x)$ such that $\sigma \in U_{i,x,e}$. $g(x)$ may only make mind changes if $U_{i,x,e}$ has not been stabilized yet. That is, $f(r(x,e),s)$ reached the least stage t such that it has changed exactly $2i{+}1$ many times. Hence, the number of mind changes g can make is bounded by a computable function. Therefore it is ω-c.e.

Now we prove that each $J^\sigma(x) \leq g(x)$. Suppose that $J^A(x) \downarrow$, let σ be the shortest initial segment in which J^A halts on. Then $r(x,e) \in A'$, as if not, we have $f(\varphi_e(x),s){=}f(r(x,e),s) = 0$ at some stage s, so we put σ into $U_{i,x,e}$ for some i. But then $J^A(r(x,e)) \downarrow$ so $r(x,e) \in A'$, which is a contradiction.

(ii) Suppose that A is a c.e. set, $g(x)$ a ω-c.a. function that dominates $J^A(x)$ for all x such that $J^A(x) \downarrow$. So $g(x)$ can be approximated by some $g(x,s)$ with a computably bounded number of mind changes. Let c be a computable function such that $c(x)$ is the least y such that $J^A(y) \downarrow$ and $J^A(y)$ greater than the least s such that $J^A(x)[s] \downarrow$. We aim to build a function p such that $A'(x) = \lim_s p(x,s)$ and has a computably bounded number of mind changes. Define p as follows:

$$p(x,s) = \begin{cases} 1 & \text{if } J^A(x)[g(c(x),s)] \downarrow \\ 0 & \text{otherwise} \end{cases}$$

Notice that $s = J^A(c(x)) \leq g(c(x),s)$. So $x \in A'$ iff there is some s such that $J^A(x)[g(c(x),s)] \downarrow$. Hence $A'(x){=}\lim_s p(x,s)$. It is also easy to see that the number of $p(x,s)$ changes can not exceed that of $g(c(x),s)$, since $p(x,s)$ may make a mind change only if $g(c(x),s)$ has made a mind change and it is greater than all $g(c(x),t)$, $t < s$. So A is superlow. $\square$

The negation of this second property implies High(MLR, weak Demuth). Thus if a c.e. set A is not superlow, Ω^A is weakly Demuth random (a set is *weakly Demuth random* if it passes all Demuth tests such that $[W_{h(0)}] \supseteq [W_{h(1)}] \supseteq [W_{h(2)}]....$) and cups A above $\emptyset'$. Therefore the following should be enough to complete the second implication:

Proposition 5.13. *If A is not ω-c.a.-jump dominated then A is High(ML-random,weakly Demuth random).*

Proof. Suppose A is not High(ML random, weakly Demuth random). Let Z be a ML-random set relative to A and fix a weak Demuth test $(G_m)_{m\in\mathbb{N}}$. Since Z is not weakly Demuth random, $Z \in G_m$ for every m. Let f be the ω-c.a. function such that $[W_{f(m)}]^\prec = G_m$ for all m. The idea to show that there is a ω-c.a. function g that dominates each J^A.

Fix m. Let $0 = s_0 < s_1 ... < s_N$ list all s such that $f(m,s) \neq f(m,s-1)$. To construct g, first define a oracle A-Solovay test $(S_m)_{m\in NN}$ as follows:

At stage t, let i be the largest stage s_i such that $s_i \leq t$. For each initial segment σ of A such that $J^\sigma(m)$ converges for the first time at stage t, put σ into an auxiliary set C_i and put each $[\tau] \subseteq [W_{f(m,s_i)}]^\prec$ into S^A_m. The total weight of the strings enumerated into S^A_m is at most 2^{-m} for each m. Thus for almost all m, we have $Z \notin S^A_m$.

Let $g(m) = \max\{J^\sigma(m) : \sigma \in C_i \wedge i < N\}$. Clearly this is a ω-c.a. function, as the number of mind changes may not exceed N. For any m, if $J^A(m)\downarrow$, then the first stage at which $J^A(m)$ converges must be less than s_N, where N is as above, as otherwise we would have $Z \in G_m{=}[W_{f(m,s_N)}]^\prec = [W_{f(m)}]^\prec \subseteq S^A_m$. Which is a contradiction. So it follows that $J^A(m) \leq g(m)$. $\square$

Proposition 5.12 shows that if a c.e set A is not superlow then it is not ω-c.a. jump dominated. The above implication shows that A would also be High(ML-random,weakly Demuth Random). For every set A, we have $\emptyset' \leq_T A \oplus \Omega^A$. So every none superlow c.e. set A is weak Demuth cuppable.

6. Connecting randomness and tracing

The two tracing classes defined in definition 2.12 and 2.11 relate nicely to highness for pairs of randomness notions by results in [11, 3]. Recall that for randomness notions $\mathcal{C} \supset \mathcal{D}$, we let High$(\mathcal{C}, \mathcal{D})$ be the class of oracles A such $\mathcal{C}^A \subseteq \mathcal{D}$ (A is strong enough to push $\mathcal{C}$ inside $\mathcal{D}$). These are said to be the double highness properties.

In this section we investigate the strength of being ω-c.a. and Δ^0_2 tracing. The work done in [12, 4] gave us a good idea of how these properties can be characterized in terms of randomness notions. The following is one of the discoveries made in the paper:

Theorem 6.1. *Let A be an oracle.*

(a) *$A \in \text{High}(\mathsf{MLR}, \mathsf{Demuth}) \Leftrightarrow A$ is ω-c.a.-tracing.*
(b) *$A \in \text{High}(\mathsf{MLR}, \mathsf{SR}[\emptyset']) \Leftrightarrow A$ is Δ^0_2 tracing.*

Before we begin to give a proof we need to establish the following lemma:

Lemma 6.2. *Suppose $\{Z \mid \exists^\infty n, Z \in U_n\} \subseteq R$ for open sets U_n,R with $\lambda(R) < q < 1$. Then there is a string τ and $d \in \mathbb{N}$ such that*

$$\lambda_\tau(R) < q \text{ and } \forall n > d[\lambda_\tau(U_n - R) = 0]$$

Proof. Assume for a contradiction the conclusion fails. Define inductively a sequence of strings $(\tau_d)_{d\in\mathbb{N}}$ such that $\tau_0 \prec \tau_1 \prec ....$ and $\forall d\ \lambda(R \mid \tau_d) < q$.

Without loss of generality we may start with τ_0 being the empty string. Suppose τ_d has been defined and $\lambda(R \mid \tau_d) < q$. If the lemma fails, there is a $n > d$ such that $\lambda(U_n - R)$. So we choose y such that $[y] \subseteq U_n$ and $\lambda([y] - R \mid \tau_d) > 0$; in particular, $y \succcurlyeq \tau_d$. By the Lebsegue density theorem we may choose $\tau_{d+1} \prec y$ such that $\lambda(R \mid \tau_{d+1}) < q$.

Now let $Z = \bigcup_d \tau_d$; then $\exists^\infty n$ such that $Z \in U_n$ and $Z \notin R$, contradiction. This establishes the lemma. □

The proof for (a) and (b) are very similar, we give the Demuth case first then show how the idea can be translated to the SR$[\emptyset']$ case.

Proof. ($\Leftarrow$): Suppose A is ω-c.a.-tracing. Fix a Demuth test $(G_m)_{m\in\mathbb{N}}$. We aim to cover this test with an A-Solovay test. Let $(T^A_m)_{m\in\mathbb{N}}$ be a c.e trace relative to A that traces f. By the Terwijn and Zambella result, the trace bound can be replaced with any arbitrary order function. Hence choose our order function to be m, and force $|T^A_m| \leq m$.

Next, for each m, let the components of T^A_m contain the least s such that $f(m)$=$f(m, s)$ i.e. the least stage that the function f stabilizes on input m. Note that since $(G_m)_{m\in\mathbb{N}}$ is a Demuth test, by definition $G_m = [W_{f(m)}]^\prec$ for some ω-c.a. function f for all m.

Build an A-Solovay test $(S^A_m)_{m\in\mathbb{N}}$ as follows: for each $s \in T^A_m$, enumerate $[W_{f(m,s)}]^\prec$ into S^A_m, eventually $f(m, s)$ stabilizes and $[W_{f(m,s)}]^\prec = [W_{f(m)}]^\prec$. G_m is a Demuth test, so each $[W_{f(m)}]^\prec$ has measure at most 2^{-m},

$\sum_m \lambda(S^A_m) \leq \sum_m m2^{-m} < \infty$, hence we conclude that S^A_m satisfies the conditions for being a Solovay test. By this construction, we have defined a Solovay test that covers all arbitrary Demuth tests. No set that is in infinitely many S_m is ML-random relative to A. Therefore $A \in \mathrm{High}(\mathsf{MLR}, \mathsf{Demuth})$.

($\Rightarrow$): Given an arbitrary ω-c.a. function f, we will define a A-c.e. trace $(T^A_n)_{n\in N}$ for f with bound 2^n.

It suffices to build an A-c.e. trace for the function given by $g(n) := nf(n) + n$, as if $(T^A_n)_{n\in N}$ traces $g(n)$, and we would like to trace $f(n)$, simply define a new A-c.e trace $(T^A_x)_{x\in N}$ by dividing each element in T^A_x by x then taking away 1 from it. $(T^A_n)_{n\in N}$ will retain the same bound, and traces $f(x)$. (Notice that $g(n)$ is divisible by n, this will become very important later on).

Next we provide detail on how to code information about g into a Demuth test relative to A. For each $n, k \in \mathbb{N}$, let

$$B_{k,n} = [\{x0^n : |x| = k\}]^{\prec}$$

Thus $B_{n,k}$ is the set of reals that have n consecutive 0s starting at the k-th digit and $\lambda(B_{k,n}) = 2^{-n}$ for all k,$n \in \mathbb{N}$.

Let $U_n = \bigcup_{n>d} B_{g(n),n}$. This is a Demuth test as $\lambda U_n = \sum_{n>d} \lambda B_{g(n),n} \leq 2^{-n}$. As $U_n = \bigcup_{n>d}[\{x0^n : |x| = g(n)\}]^{\prec}$ and $g(n)$ is a ω-c.a. function, $U_n = [W_{g'(n)}]^{\prec}$ for some ω-c.a. function g'.

Now let R be the second member of the universal ML test relative to A, so that $\lambda(R)< 2^{-2}$. Since $\mathsf{MLR}[A] \subseteq \mathsf{Demuth}$, $\bigcap_d U_d \subseteq R$. By Lemma 6.2 there is a string τ and $d \in \mathbb{N}$ such that $\lambda_\tau(R) < 2^{-2}$ and $\lambda_\tau(B_{g(n),n} - R) = 0$ for all $n > d$. Now let $n\mathbb{N}$ denote the multiples of n and consider the following trace:

$$T_n = \{k \in n\mathbb{N} \mid \lambda_\tau(B_{k,n} - R) < 2^{-k-3}\}.$$

Since $B_{k,n}$ is clopen and R is $\Sigma^0_1[A]$, the sequence (T_n) is uniformly c.e. in A. On the other hand, g is defined to be divisible by n, hence $g(n) \in T_n$ for all but (finitely) at most d many n, by the choice of d, τ.

It remains to show that $|T_n|$ is computably bounded for all n. Note that

$$\lambda_\tau(\textstyle\bigcup_{k\in T_n} B_{k,n} - R) \leq \sum_{k\in T_n} \lambda_\tau(B_{n,k} - R) < 2^{-2}$$

which implies that

$$\lambda_\tau(2^\omega - \textstyle\bigcup_{k\in T_n} B_{k,n}) + \lambda_\tau(R) \geq 1 - 2^{-2}.$$

Since $\lambda_\tau(R) < 2^{-2}$, $\lambda_\tau(2^\omega - \bigcup_{k\in T_n} B_{k,n}) > 2^{-1}$. On the other hand, $\lambda_\tau(B_{k,n}) = 2^{-n}$ for $n > |\tau|$. Since T_n consists of multiples of n, the sets $B_{k,n}, k \in T_n$ are *independent*. (Suppose $N \in \mathbb{N} \cup \{\infty\}$, and let $S_k \subseteq 2^{\mathbb{N}}$ be measurable for $k < N$. In probability theory, the events S_k are called *independent* if for each finite set $Y \subseteq \{k : k < N\}$, we have $\lambda \bigcap_{k\in Y} S_k = \prod_{k\in Y} \lambda(S_k)$). Which allows us to conclude that:

$$1/2 < \lambda_\tau(2^\omega - \textstyle\bigcup_{k\in T_n} B_{k,n}) = \lambda_\tau \bigcap_{k\in T_n}(2^\omega - B_{n,k}) = (1 - 2^{-n})^{|T_n|}$$

for $n > |\tau|$. With some algebraic manipulation, set $r = 2^n - 1$, we get $(1 - 2^{-n})^{|T_n|} = (r/r + 1)^{|T_n|}$, and $2 > (r + 1/r)^{|T_n|}$. But $(r + 1/r)^x \geq 2$ for all $x > r$ as $(r+1)^r \geq r^r + r^{r-1}\binom{r}{1} = 2r^r$. Therefore $|T_n| < r < 2^n$ for $n > |\tau|$. □

Proof. ($\Leftarrow$): By Lemma 4.2 it suffices to cover every limit test with a A-Solovay test. Fix a limit test $(V_m)_{m\in\mathbb{N}}$. Now each $V_m = [W_{f(m)}]^{\prec}$ for some $f <_T \emptyset'$. As before, we build the same trace $(T^A_m)_{m\in\mathbb{N}}$ and A Solovay test $(S^A_m)_{m\in\mathbb{N}}$ as part (a) but with a Δ^0_2 function f instead of a ω-c.e. one. Again, each $[W_{f(m)}]^{\prec}$ has measure at most 2^{-m}, so $\sum_m \lambda(S^A_m) \leq \sum_m m2^{-m} < \infty$ $(V_m)_{m\in\mathbb{N}}$.

By the construction of the Solovay test, it contains all arbitrary limit tests, and the result follows.

Since we showed that $\mathsf{SR}[\emptyset']$ is in fact the same as limit random, we shall give the proof for (b) in a different way. Instead of the original proof, which used $\emptyset'$ as an oracle to identify components during the construction, we use the fact that a limit test is just a Demuth test where the function h used to determine its members is not ω-c.e. but Δ^0_2. This allows us to slightly modify the above proof to reach the desired result:

($\Rightarrow$): Suppose f is an Δ^0_2 function and we wish to build an A-c.e trace for f with bound 2^n. As before, we instead build an A-c.e. trace for the function by $g(n) = nf(n)+n$. Let $U_n = B_{g(n),n}$. Since g is Δ^0_2, the sequence $(U_n)_{n\in\mathbb{N}}$ forms a limit test. As before, let R be the second member of the universal ML test relative to A, so that $\lambda(R) < 2^{-2}$. By our assumption we have $\mathsf{MLR}[A] \subseteq \mathsf{LR}$, we may pick τ,d according to Lemma 6.2 where $q = 2^{-2}$. Define the same A-c.e. trace:

$$T_n = \{k \in n\mathbb{N} \mid \lambda_\tau(B_{k,n} - R) < 2^{-k-3}\}.$$

By the lemma we have $\lambda_\tau(R) < q$ and $\forall n > d[\lambda_\tau(U_n - R{=}0]$. Hence , as before, $|T_n| < 2^n$ and $(T_n)_{n\in\mathbb{N}}$ is a trace for g. □

6.1. **Variations on traceability notions.**

We again begin this section with the necessary definitions:

Definition 6.3. Let A be an oracle in 2^ω, we say that a function f is *bounded limit recursive* by A (abbreviated to $\mathbf{BLR}\langle A\rangle$) if there is a uniformly A-computable sequence of functions f_s converging to f, such that the number of mind changes $\#\{s : f_{s+1}(n) \neq f_s(n)\}$ is bounded by a computable function.

Its also easy to see that f is $\mathbf{BLR}\langle A\rangle$ iff its computable in A' with unbounded A-use but computably bounded A' use.

Definition 6.4. Let A be an oracle, a Demuth$_{BLR}\langle A\rangle$-*test* is a sequence of $\{U^A_n\}_{n\in\mathbb{N}}$ A-c.e. subsets of 2^ω such that for each n, $\lambda(U^A_n) \leq 2^{-n}$, and there is a $\mathbf{BLR}\langle A\rangle$-function taking n to a A-c.e.-index for U^A_n.

We say that a set is Demuth$_{BLR}\langle A\rangle$ random if it passes all Demuth$_{BLR}\langle A\rangle$-*tests*.

Notice that the difference between $\text{Demuth}_{BLR}\langle A\rangle$-tests and Demuth^A tests is that, for the former, the number of changes that a component U_n^A can make is no longer bounded by an A-computable function but a computable one. Furthermore, if we set $A = \emptyset'$, we obtain the Demuth random sets.

Definition 6.5. An oracle A is *Demuth traceable* if there is an order function h, such that every $\mathbf{BLR}\langle A\rangle$ function has an trace T_n bounded by h and there is an ω-c.a. function taking n to a c.e. index for T_n.

R. Downey, N. Greenberg and A. Nies investigated Demuth traceability in [4]. For instance, superlow c.e. sets sets are Demuth traceabile.

They also observed that for any set A:

Demuth traceable $\Rightarrow$ Demuth noncuppable $\Rightarrow$ not ω-c.a.-tracing.

The second implication follows from Theorem 6.1 by taking Ω^A. Suppose A is ω-c.a.-tracing, then $A \in \mathsf{High}(\mathsf{MLR}, \mathsf{Demuth})$. So $\mathsf{MLR}^A \subseteq \mathsf{Demuth}$, and since $\Omega \in \mathsf{MLR}$, $\Omega^A \in \mathsf{Demuth}$. We know that Ω is Turing-complete and so $\emptyset' \leq_T A \oplus \Omega^A$ for any set A. Hence A is Demuth cuppable.

The first implication was first proved in [4]. They used the fact shown in [[21] Thm 3.6.26], the construction shows that if Z is a Demuth random set, then there is a ω-c.a. function f dominating the jump of Z, hence Z must be GL_1, i.e. $Z' \leq_T Z \oplus \emptyset'$. Adding in the full details not present in the original paper we obtain the following:

Theorem 6.6. *Suppose A is Demuth traceable. Then A is not Demuth cuppable.*

Proof. Partially relativizing the result in [21, Thm 3.6.26] we aim to show that $A \oplus Y \not\geq_T \emptyset'$ for each Demuth random set Y. In [4] it is shown that each Demuth random set is $\text{Demuth}_{BLR}\langle A\rangle$ random. So we need to construct a function f $\mathbf{BLR}\langle A\rangle$ dominating $\Gamma^{A\oplus Y}$.

To do this first we need to define a Turing functional Γ by

$$\Gamma^{A\oplus Y}(m) \simeq \mu s.\, \Phi_m^{A\oplus Y}(m)[s]\downarrow$$

For f to dominate we require $\forall^\infty m(\Gamma^{A\oplus Y}(m)\downarrow \to \Gamma^{A\oplus Y}(m) \leq f(m))$.

To build the function f we do so via a construction at stages. First we let $\mathcal{P}_m$ be the set of ms the Turing functional halts on, i.e. $\mathcal{P}_m = \{Y : \Phi_m^{A\oplus Y}(m)\downarrow\}$ and $\mathcal{P}_{m,s} = \{Y : \Phi_{m,s}^{A\oplus Y}(m)\downarrow\}$ be the approximation at stage s.

The construction results in two items, the function f and an sequence of auxiliary clopen sets $\mathcal{Q}_{m,s}$ that contains all the oracles Y such that at every stage of the construction, we have the desired domination.

Formal construction.

At stage 0, let $\mathcal{Q}_{0,0} = \emptyset$ and $f_0(0) = 0$

At stage s, reset both $\mathcal{Q}_{s,s}$ and $g_s(s)$ to its value during stage 0. For each $m > s$, let $g_{m,s}(m)$ and $Q_{m,s}$ be the same as stage $s-1$, i.e. $g_{m,s}(m) = g_{m,s-1}(m)$ and $\mathcal{Q}_{m,s} = \mathcal{Q}_{m,s-1}$. Otherwise, first we check if the clopen set $\mathcal{P}_{m,s} - \mathcal{Q}_{m,s-1}$ exceeds 2^{-m}. If $\lambda(\mathcal{P}_{m,s} - \mathcal{Q}_{m,s}) > 2^{-m}$ set $\mathcal{P}_{m,s} = \mathcal{Q}_{m,s}$, and $f_s(m) = s$ (this number is sufficiently large enough to dominate $\Gamma^{A\oplus Y}(m)$ since all computation at stage s does not exceed s).

End of construction.

This construction gives us a Demuths test $(U_m)_{m\in\mathbb{N}}$ for each Y that passes S_m. If Y is Demuth random then $Y \notin U_m$ for almost all m, so f dominates $\Gamma^{A\oplus Y}$.

We can verify that f is indeed $\mathbf{BLR}\langle A\rangle$ as whenever a new oracle is added, we change the auxiliary set $\mathcal{Q}_m$ and increase f. These increases to $f(m)$ can happen at most 2^m times, and $\mathcal{Q}$ eventually stabilises. Therefore our construction gives use the necessary $U_m = \mathcal{P}_m - \mathcal{Q}_m$ to form a Demuth test. Hence, if Y is Demuth random then $Y \notin U_m$ for almost all m, so f dominates $\Gamma^{A\oplus Y}$.

Lastly, since A is Demuth traceable, f has a Δ^0_2 upper bound. Hence $A \oplus Y \not\geq_T \emptyset'$. □

7. Tracing for variants of the class of $\omega - c.a.$ functions

The following definition is derived by generalizing Definitions 2.12 and 2.11. The intuition is that we will like to investigate traceability without constantly referring to the Terwijn and Zambella result.

Definition 7.1. Let $\mathcal{C}$ be a countable class of total functions on $\mathbb{N}$, $f : \mathbb{N} \to \mathbb{N}$.

• A *$\mathcal{C}$-Demuth test* is a sequence of open cylinders $[W_{f(m)}]^{\prec}_{m\in\mathbb{N}}$ where f is in the class $\mathcal{C}$.

• We say that Z is *$\mathcal{C}$-Demuth random* if for each *$\mathcal{C}$-Demuth tests* $[W_{f(m)}]^{\prec}_{m\in\mathbb{N}}$ we have $Z \in [W_{f(m)}]^{\prec}$ for all but finitely many n.

• We call a set *A $\mathcal{C}$-tracing with bound h* if each function in $\mathcal{C}$ has a A-c.e trace $(T^A_x)_{x\in\mathbb{N}}$ such that $|T^A_x| \leq h(x)$ for each x.

Notice that the trace bound 2^x has been replaced with some arbitrary order function $h(x)$.

With these definitions in mind, its easy to imagine that Theorem 4.2 can be generalised in some way in terms of $\mathcal{C}$-Demuth randomness. This will be shown to be true in this section, however, the trace bound is required to be adjusted, to bounds like as $2^m m^{-2}$, in order to achieve this double highness notion.

Lemma 7.2. *A is $\mathcal{C}$ tracing with a bound g such that $\sum g(n)2^{-n} < \infty \Rightarrow$*
$$A \in \mathsf{High}(\mathsf{MLR}, \mathcal{C} - \mathsf{Demuth}).$$

Proof. Fix a $\mathcal{C}$-Demuth test $[W_{f(m)}]^{\prec}_{m\in\mathbb{N}}$. We build a Solovay test to cover each $W_{f(m)}$. Let $(T_m)_{m\in\mathbb{N}}$ be a A-c.e. trace for f with bound g. Define an A Solovay test $(\mathcal{S}^A_m)$ as follows: for each $k \in T_m$, enumerate the open set $[W_{f(k)}]^{\prec}$ into $\mathcal{S}^A_m$ as long as its measure is $\leq 2^{-k}$. $\sum_m \lambda(\mathcal{S}^A_m) \leq \sum_m g(m) < \infty$. Thus no set that is in infinitely many $\mathcal{S}^A_m$ can be ML-random relative to A. □

We now look at subclasses of ω-c.a., and also classes $\mathcal{C}$ containing ω-c.a.

For a computable order function g, let $\mathcal{C}$ be the class of functions $f \leq_{\mathrm{wtt}} \emptyset'$ such that some computable approximation for $f(x)$ has at most $g(x)$ changes. Then A is $\mathcal{C}$ tracing with bound 2^m if each function in $\mathcal{C}$ has an A-c.e. trace $(T^A_x)_{x\in\mathbb{N}}$ such that $|T^A_x| \leq 2^x$ for each x. We also say that A is *g-c.a.-tracing* with bound 2^m. More generally, we could have a class $\mathcal{D}$ of computable functions instead of a single g, and we say A is $\mathcal{D}$-c.a.-tracing with the obvious meaning.

Let Z be ML-random. By [12, Thm 23], Z is ω-c.a.-tracing iff Z is $2^n h(n)$-c.a.-tracing, where h is an arbitrary (say, slowly growing) order function.

By the Terwijn Zambella argument and the first lemmas of the section, we have:

Proposition 7.3. *Let $\mathcal{D}$ be a class of computable functions (bounds) such that for each function $f \in \mathcal{D}$, the function*
$$\hat{f}(n) = \textstyle\sum_{i\leq n+3\log n} f(i)$$

is also in $\mathcal{D}$. Let $\mathcal{C}$ be the class of $\mathcal{D}$-c.a. functions. Then $A \in \mathrm{High}(\mathsf{MLR}, \mathcal{C} - \mathsf{Demuth}) \Leftrightarrow A$ is $\mathcal{C}$-tracing with bound 2^n.

Proof. ($\Rightarrow$:) The forward direction is another modification of the proof of Theorem 4.2:

Suppose f is an $\mathcal{C}$-c.a. function and we wish to build an A-c.e. trace for f with bound 2^n. Once again we may change the function we are tracing to $g(n) := nf(n) + n$. Next, we let $U_n = B_{g(n),n}$ as in Theorem 6.2. Since g is computable in $\mathcal{C}$, the sequence of $(U_n)_{n\in\mathbb{N}}$ is a $\mathcal{C}$-Demuth test. Again, let R be the second universal Martin-Löf test relative to A, as before $\lambda(R) < 2^{-2}$. Since we have $\mathsf{MLR}^A \subseteq \mathcal{C} - \mathsf{Demuth}$, the conditions for applying 6.2 has been met, and we choose τ and d accordingly.

Now, define the same trace $(T_x)_{x\in\mathbb{N}}$ as in Theorem 4.2, since $\lambda_\tau(R) < q$ and $\forall n > d[\lambda_\tau(U_n - R) = 0]$. As before, each T_n has the appropriate bound, so $(T_n)_{N\in\mathbb{N}}$ is a trace for g. Hence we can derive a trace for f with the same technique mentioned in Theorem 4.2.

Unfortunately, we can not achieve the converse the a direction modification. Instead we have to do use the fact shown in lemma 7.2 and do a little more work:

($\Leftarrow$): By 7.2 we just need to find a bound g that satisfies the conditions $\sum g(n)2^{-n} < \infty$. If A is $\mathcal{C}$ tracing with bound g, then $A \in \mathrm{High}(\mathsf{MLR}, \mathcal{C} - \mathsf{Demuth})$ and we are done.

We use the same representation approach to constructing g as the Terwijn Zambella argument in Theorem 2.17. Let $q(n) = \max\{i : i{+}3 \log i \leq n\}$ and let $g(n) = 2^{q(n)+1}$. For each function $h \in \mathcal{D}$, the function $\hat{h}(i) = h \upharpoonright_{i+3\mathrm{log}i}$, i.e. the function that maps i to the tuple of the first $i + 3 \log i$ values of h, is also in $\mathcal{C}$.

Now we have to check that $\sum g(n)2^{-n}$ is indeed finite. Let

$$f(n) := n - 2\mathrm{log}n,$$

notice that for any n we have:

$$q(n) = \max\{i : i + 3 \log i \leq n\} < n - 3\mathrm{log}(i) < n - 3\mathrm{log}(n).$$

Since $2^{r(n)-n} = n^{-2}$ and $q(n) \leq r(n)$. $\sum g(n)2^{-n} < \sum n^{-2} < \infty$. $\square$

Note that $(n{+}3) \log n$ is not the lowest the bound could be, originally we chose $n\mathrm{log}^2 n$ for its attractive convergent property, this would have, however, yielded a lesser result. Finding a lower bound is currently unknown.

An example of such a $\mathcal{D}$ is the class of computable functions bounded by a function of type $2^n \cdot q(n)$ where q is a polynomial.

7.0.1. *Classes $\mathcal{C}$ containing ω-c.a.* Due to the lack of a better word, the following shall be referred to as the $(*)$ property.

Definition 7.4. We say a class $\mathcal{C}$ is $(*)$ closed if for have any computable function p and each $f \in \mathcal{C}$, the mapping function $f^* : x \to f \upharpoonright_{p(x)}$ also in the class $\mathcal{C}$.

Its natural to ask what type of functions satisfies this property. Through some rigorous investigation with Andre, we discovered that, if $R = (\mathbb{N}, <_R)$

is a computable well-order of type ω^n. Then the class of functions with ω^n approximations has this property.

Definition 7.5. An ω^n-approximation is a computable function
$g = \langle g_0, g_1\rangle : \mathbb{N}\times\mathbb{N} \to \mathbb{N}\times\mathbb{N}$
such that for each input x and stage $s > 0$, we have
$g(x,s) \neq g(x,s-1) \to g_1(x,s) <_R g_1(x,s-1)$.
In this setting, if g_0 is a computable approximation of a total Δ^0_2 function f, we say that g is an ω^n-approximation of f.

We give a fine analysis of Theorem 6.1 in this more general setting. We show the double highness notion $A \in \mathrm{High}(\mathsf{MLR}, \mathcal{C}-\mathsf{Demuth})$ implies A is $\mathcal{C}$-tracing at bound 2^m. However, we need tracing at slightly better bound, such as $2^m m^{-2}$, to reobtain the double highness notion. Thus there is a gap in the absence of condition $(*)$.

The second part is a modification of the proof of the corresponding result [11, Prop. 32]; also see [3, Thm. 3.6].

Intuitively speaking, for an ω^n approximation $g = \langle g_0, g_1\rangle$, we may think of g_1 as an n-tuple of computable functions $\langle g_{11}, g_{12}, g_{13}, ... g_{1n}\rangle$. At stage s, $g_0(x,s)$ serves as the value approximation of g and $g_1(x,s)$ as a counter to the number of possible changes.

Theorem 7.6. *The class of ω^n-c.a. functions satisfies the $(*)$ condition.*

The proof is an extension of [22] where $n = 2$. We expand it to the more general ω^n case.

Proof. Fix some computable function p, let $\widehat{f}\,(x) = f\restriction_{p(x)}$ i.e. the function that maps x to the tuple of the first $p(x)$ values of f, encoded by some natural number.

If we can show that f has an ω^n-approximation iff $\widehat{f}$ has an ω^n-approximation then we are done, as it would mean that ω^n is closed under alterations on initial segments by computable functions.
($\Rightarrow$:) Let $\langle g_0, g_1\rangle$ be an ω^n approximation of f. As we discussed before, we view g_1 as a n-tuple of computable functions $\langle g_{11}, g_{12}, g_{13}, ... g_{1n}\rangle$.

Let $h_0(x,s) = g_0(x,s)\restriction_{p(x)}$ and $h_{1i}(x,s) = \sum_{y<p(x)} g_{1i}(y,s)$

Now we show that $\langle h_0, \langle h_{11}, h_{11}, h_{12}, ... h_{1n}\rangle\rangle$ is an ω^n-approximation.

First notice that each $h_{1i}(x,s)$ is non-increasing in s. Suppose at stage s, $k < n$ is the least k such that $\exists y < p(x)[g_{1k}(y,s) \neq g_{1k}(y,s-1)]$. Since g_1 is a ω^n-approximation, we have $g_{1k}(y,s) < g_{1k}(y,s-1)\dot{\text{N}}$ow consider y' such that $y' < p(x)$, by the minimality of k, we must have $g_{1k}(y',s) \leq g_{1k}(y',s-1)$. Thus $h_{1k}(x,s) < h_{1k}(x,s-1)$ and also $h_{1i}(x,s) = h_{1i}(x,s-1)$ for each $i < k$.

If $h_0(x,s) \neq h_0(x,s-1)$ then there must be a least k and y such that $g_{1k}(y,s) \neq g_{1k}(y,s-1)$. Whence $h_1(x,s) < h_1(x,s-1)$.
($\Leftarrow$:)Let $\langle h_0, h_1\rangle$ be an ω^n approximation of $\widehat{f}$. Construct g_0 as follows:

$$g_0(x,s) = \begin{cases} h_0(x,s) & \text{if } h_0(x,s) \text{ encodes a tuple of length } x+1 \\ 0 & \text{otherwise} \end{cases}$$

Let $g_1(x,s) = h_1(x+1,s)$. If at stage s, $g_0(x,s) \neq h_0(x,s)$ then the length of $h_0(x,s)$ must be wrong, hence we reset $g_0(x,s) = 0$. The counter $g_1(x,s)$

is defined to be $h_1(x+1,s)$, which means it counts down in ω if and only if $h_1(x+1,s)$ does. Hence, $\langle g_0, g_1\rangle$ is an ω^n-approximation of f. $\square$

We conclude the thesis with a conjecture, which I believe to be true, that for c.e. sets A, one may ask if there is a proper hierarchy of being g-c.a.-tracing, for faster and faster growing computable functions g:

Conjecture 7.7. *Let g be computable. Is there a computable function h and a c.e. set A that is g-c.a.-tracing, but not h-c.a.-tracing?*

Sketch of the proof: Modify the proof of Proposition 3.3. Here we instead build a g-c.a.-tracing c.e.-set A that is superlow. Hence this set is not h-c.a.-tracing for an appropriate faster growing h. The positive requirement will be adjusted so that $(T_x^A)_{x\in\mathbb{N}}$ will instead be a g-c.e. oracle trace and we will have to make sure that the number of mind changes of $\Phi_e^A(e)$ is always bounded by some computable function.

References

[1] A.Wald. Sur le notion de collectif dans la calcul des probabilities. 1936. 1.1

[2] G. Barmpalias. Tracing and domination in the Turing degrees. *Ann. Pure Appl. Logic*, 2011. To appear. 3.9, 3.1

[3] G. Barmpalias, J. Miller, and A. Nies. Randomness notions and partial relativization. *Israel J. Math.*, 2011. In press. 6, 7.0.1

[4] L. Bienvenu, R. Downey, N. Greenberg, A. Nies, and D. Turetsky. Lowness for Demuth randomness. Unpublished, 20xx. 6, 6.1, 6.1

[5] L. Bienvenu, R. Hoelzl, J. Miller, and A. Nies. The Denjoy alternative for computable functions. In *STACS 2012*, 2012. 5

[6] Adam R. Day and Joseph S. Miller. Cupping with random sets. To appear in Proc.AMS. 5, 5.3

[7] O. Demuth. Remarks on the structure of tt-degrees based on constructive measure theory. *Comment. Math. Univ. Carolin.*, 29(2):233–247, 1988. 2.6

[8] R. Downey and E. Griffiths. Schnorr randomness. *J. Symbolic Logic*, 69(2):533–554, 2004. 2.16

[9] R. Downey and D. Hirschfeldt. *Algorithmic randomness and complexity.* Springer-Verlag, Berlin, 2010. 855 pages. 2.9

[10] R. G. Downey, Carl G. Jockusch, Jr., and M. Stob. Array nonrecursive sets and multiple permitting arguments. In K. Ambos-Spies, G. H. Muller, and Gerald E. Sacks, editors, *Recursion Theory Week, Oberwolfach 1989*, volume 1432 of *Lecture Notes in Mathematics*, pages 141–174, Heidelberg, 1990. Springer–Verlag. 3.8

[11] S. Figueira, D. Hirschfeldt, J. Miller, Selwyn Ng, and A Nies. Counting the changes of random Δ^0_2 sets. In *CiE 2010*, pages 1–10, 2010. Journal version to appear in J.Logic. Computation. 1.2, 3, 3, 6, 7.0.1

[12] S. Figueira, D. Hirschfeldt, J. Miller, Selwyn Ng, and A Nies. Counting the changes of random Δ^0_2 sets. pages 1–10, 2012. Journal version to appear in J.Logic. Computation. 3.1, 5, 6, 7

[13] S. Figueira, A. Nies, and F. Stephan. Lowness properties and approximations of the jump. *Ann. Pure Appl. Logic*, 152:51–66, 2008. 1.2

[14] Johanna N. Y. Franklin and Keng Meng Ng. Difference randomness. *Proc. Amer. Math. Soc.*, 139(1):345–360, 2011. 5

[15] N. Greenberg and A. Nies. Benign cost functions and lowness properties. *J. Symbolic Logic*, 76:289–312, 2011. 2.2

[16] A. N. Kolmogorov. Three approaches to the definition of the concept "quantity of information". *Problemy Peredači Informacii*, 1(vyp. 1):3–11, 1965. 2.2

[17] L. A. Levin. The concept of a random sequence. *Dokl. Akad. Nauk SSSR*, 212:548–550, 1973. 2.2

[18] P. Martin-Löf. The definition of random sequences. *Inform. and Control*, 9:602–619, 1966. 1.1, 2.3, 2.14

[19] J. Miller and A. Nies. Randomness and computability: Open questions. *Bull. Symbolic Logic*, 12(3):390–410, 2006. 5

[20] A. Nies. Reals which compute little. In *Logic Colloquium '02*, Lecture Notes in Logic, pages 260–274. Springer–Verlag, 2002. 1.2

[21] A. Nies. *Computability and randomness*, volume 51 of *Oxford Logic Guides*. Oxford University Press, Oxford, 2009. 2, 2.2, 2.2, 2.3, 2.3, 2.3, 3.2, 6.1, 6.1

[22] A. Nies. Computably enumerable sets below random sets. *Ann. Pure Appl. Logic*, 163(11):1596–1610, 2012. 7.0.1

[23] C.P. Schnorr. *A unified approach to the definition of a random sequence.* Springer-Verlag, Berlin-New York, 1971. Mathematical Systems Theory. 2.4, 2.7

[24] C.P. Schnorr. *Zufälligkeit und Wahrscheinlichkeit. Eine algorithmische Begründung der Wahrscheinlichkeitstheorie.* Springer-Verlag, Berlin, 1971. Lecture Notes in Mathematics, Vol. 218. 2.2, 2.13

[25] S.Ishmukhametov. Weak recursive degrees and a problem of spector. *Recursion Theory and Complexity de Gruyter*, 66:81–88, 1997. 1.2

[26] Robert I. Soare. *Recursively Enumerable Sets and Degrees.* Perspectives in Mathematical Logic, Omega Series. Springer–Verlag, Heidelberg, 1987. 3, 3.2

[27] R. Solovay. Handwritten manuscript related to Chaitin's work. IBM Thomas J. Watson Research Center, Yorktown Heights, NY, 215 pages, 1975. 2.8, 2.15

[28] S.terwijn. *Computability and Measure.* Ph.D. Thesis, University of Amsterdam, 1998. 1.2

[29] S. Terwijn and D. Zambella. Algorithmic randomness and lowness. *J. Symbolic Logic*, 66:1199–1205, 2001. 1.2

[30] R. von Mises. Grundlagen der Wahrscheinlichkeitsrechnung. *Math. Zeitschrift*, 5:52–99, 1919. 1.1

[31] Liang Yu. Characterizing strong ml randomness. *Ann. Pure Appl. Logic*, 163(3):214–224, 2012. 4

[32] Domenico Zambella. On sequences with simple initial segments. ILLC technical report ML 1990-05, Univ. Amsterdam, 1990. 2.10