CS 750 Slides Week 12: Interactions of randomness and computability

André Nies

The University of Auckland

CS750, 2010, Week 12

A two-way interaction:

Randomness

interacts with

Computability

Brief Outline

Studying randomness via computability

First main point

Mathematical notions of randomness can be defined and studied using algorithmic methods.

Results:

- In contrast to the setting of probability theory, it makes sense to say that an individual object is random.
- There is not a single algorithmic randomness notion. Rather, these notions form a hierarchy.
- Randomness of a real $z \in [0,1]$ is equivalent to differentiability of certain computable functions $f\colon [0,1] \to \mathbb{R}$ at z.

Studying computational lowness via randomness

Intuitively speaking, an object (such as a real, a set of natural numbers, or a function) is low if it is close to being computable.

Second main point

Lowness properties can be defined and studied via randomness.

For instance, in sense to be specified,

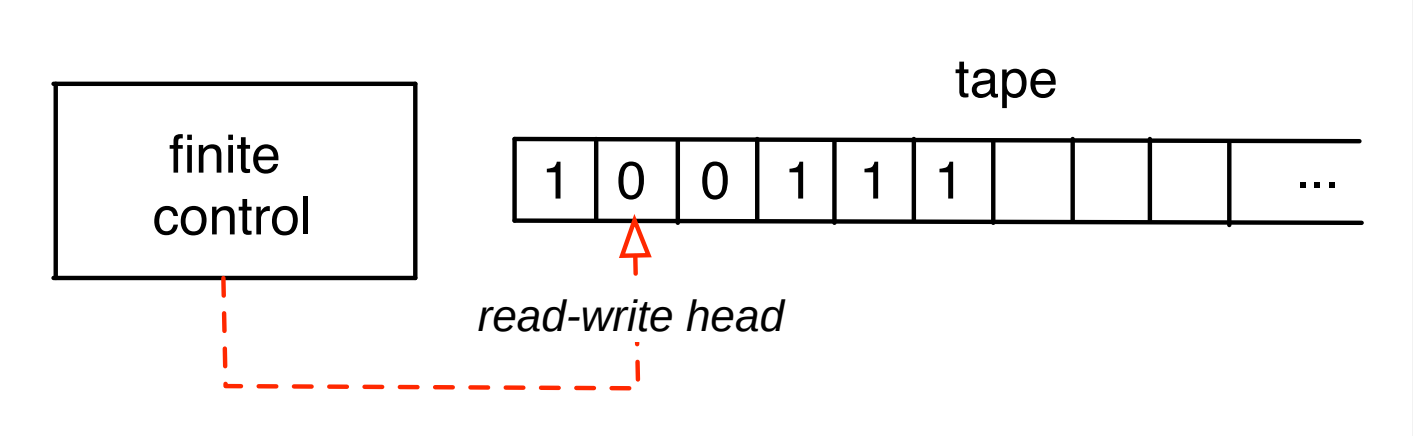
being close to computable is equivalent to being far from random.

Part I

Studying randomness via computability

Basics of computability theory 1

A Turing machine in action looks like this:



The finite control holds a Turing program.

A function $F\colon \mathbb{N} \to \mathbb{N}$ is called computable if there is a Turing program which, beginning with n in binary on the tape, ends with $F(n)$ in binary on the tape:

$$n \longrightarrow \boxed{\text{Turing program}} \longrightarrow F(n)$$

Computable reals

In the definition of computable function, $\mathbb{N}$ can be replaced by domains that are effectively encoded by natural numbers, such as the rationals $\mathbb{Q}$.

- A real $r \in \mathbb{R}$ is computable if there is a computable sequence $(q_n)_{n\in\mathbb{N}}$ of rational numbers such that $|r - q_n| < 2^{-n-1}$ for each n.
- Examples of computable reals are $\sqrt{2}, \pi, e, \ldots$

1. **Studying randomness via computability**
   - Algorithmic randomness
   - Characterizations via differentiability

Randomness via probability theory

Imagine we toss a fair coin repeatedly. This is modelled as follows.

- We have a sequence $(X_n)_{n\in\mathbb{N}}$ of 0, 1-valued "random variables" on a probability space $(M, \mathcal{B}, P)$.
- The X_n are independent. We have $P[X_n = 0] = 1/2$ for each n.
- Each element w of the space determines a sequence of coin tosses, where the n-th bit is $X_n(w)$.
- To say that a property holds for a "random" sequence means that the property holds with probability 1. Thus, the exceptions form a null set.
- An example of such a property is the law of large numbers.

Algorithmic randomness notions

The idea in algorithmic randomness

z is random $\iff$ z avoids each algorithmic null set.

- We have to specify what we mean by an algorithmic null set.
- For instance, having more than 3/4 zeros on arbitrarily long initial segments will be an algorithmic null set in the sense of Martin-Löf.

The probability spaces

In the following, the probability space will be either

- Cantor space $\{0,1\}^{\mathbb{N}}$ with the product topology, and the product measure, where $\{0,1\}$ is equipped with the measure such that both $0, 1$ have probability $1/2$, or
- the unit interval $[0,1]$ of reals, with Lebesgue measure.

In either case the probability measure is denoted P.

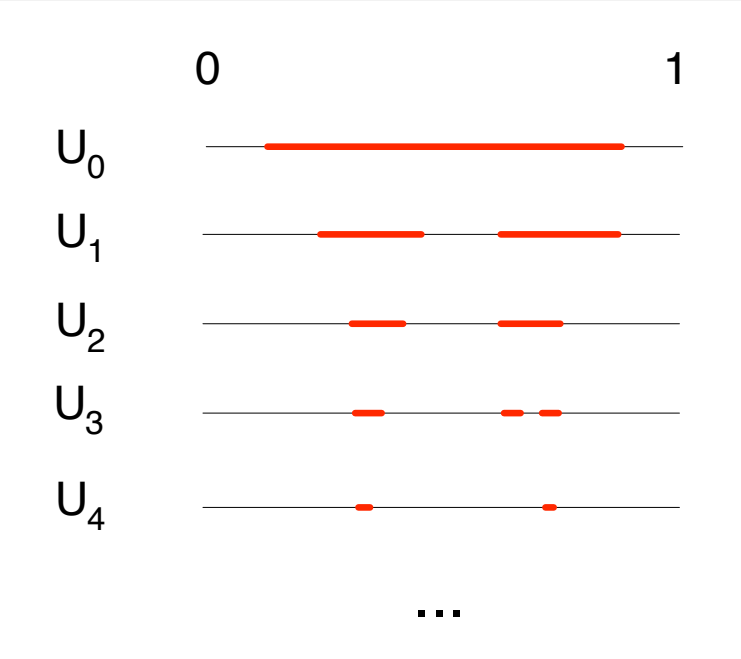
The two spaces are equivalent via the binary expansion of reals.

Algorithmic null sets in the sense of Martin-Löf (1966)

An open set $U \subseteq [0,1]$ is called computably enumerable if there is an effective list $I_0, I_1, \ldots$ of open intervals with rational endpoints such that $U = \bigcup_r I_r$.
A sequence $(U_n)_{n\in\mathbb{N}}$ of open sets is called a Martin-Löf test if

the U_n are computably enumerable, where the listing procedure has n as a second input (uniformity), and

U_n has measure at most 2^{-n} for each n.



Definition

A real r is Martin-Löf random if $r \notin \bigcap_n U_n$ for each ML-test $(U_n)_{n\in\mathbb{N}}$.

For instance, a computable real r is not ML-random:
if $(q_n)_{n\in\mathbb{N}}$ is a computable sequence of rational numbers such that $|r - q_n| < 2^{-n-1}$ for each n, let

$$U_n = (q_n - 2^{-n-1}, q_n + 2^{-n-1}).$$

Then $(U_n)_{n\in\mathbb{N}}$ is a ML-test such that $r \in \bigcap_n U_n$.

1. **Studying randomness via computability**
   - Algorithmic randomness
   - Characterizations via differentiability

Functions of bounded variation are differentiable outside a null set

A function $f:\ [0,1] \to \mathbb{R}$ is of bounded variation if it doesn't "wiggle" too much:

$$\infty > \sup \sum_{i=1}^{n-1} |f(t_{i+1}) - f(t_i)|,$$

where the sup is taken over all collections $t_1 \leq t_2 \leq \ldots \leq t_n$ in $[0,1]$.

Such an f is differentiable at a "random" real:

Theorem (Classical Analysis)

Let $f:\ [0,1] \to \mathbb{R}$ be of bounded variation. Then $f'(r)$ exists for each r outside a null set (depending on f).

Complexity of the exception set

Theorem (Demuth 1975/Brattka, Miller, Nies, to appear)

Let $r \in [0,1]$. Then
r is ML-random $\iff$
$f'(r)$ exists, for each function f of bounded variation such that $f(q)$ is a computable real, uniformly for each rational q.

- The implication "$\Rightarrow$" is an algorithmic version of the classical theorem.
- The implication "$\Leftarrow$" has no classical counterpart. To prove it, one builds a computable function f of bounded variation that is only differentiable at ML-random reals.

Randomness via betting strategies

Computable betting strategies are certain computable functions M from binary strings to the non-negative reals.

- Let Z be a sequence of bits. When the player has seen the string σ of the first n bits of Z, she can make a bet q, where $0 \leq q \leq M(\sigma)$, on what next bit $Z(n)$ is.
- If she is right, she gets q. Otherwise she loses q. Thus, we have

$$M(\sigma 0) + M(\sigma 1) = 2M(\sigma)$$

  for each string σ.
- She wins on Z if M is unbounded along Z. We call a set Z computably random if no computable betting strategy wins on Z.

Martin-Löf random $\Rightarrow$ computably random, but not conversely.

Computable randomness and differentiability

Theorem (Brattka, Miller, Nies, to appear)

Let $r \in [0,1]$. *Then*

r *is computably random* $\iff$

$g'(r)$ *exists, for each nondecreasing function* g *that is uniformly computable on the rationals.*

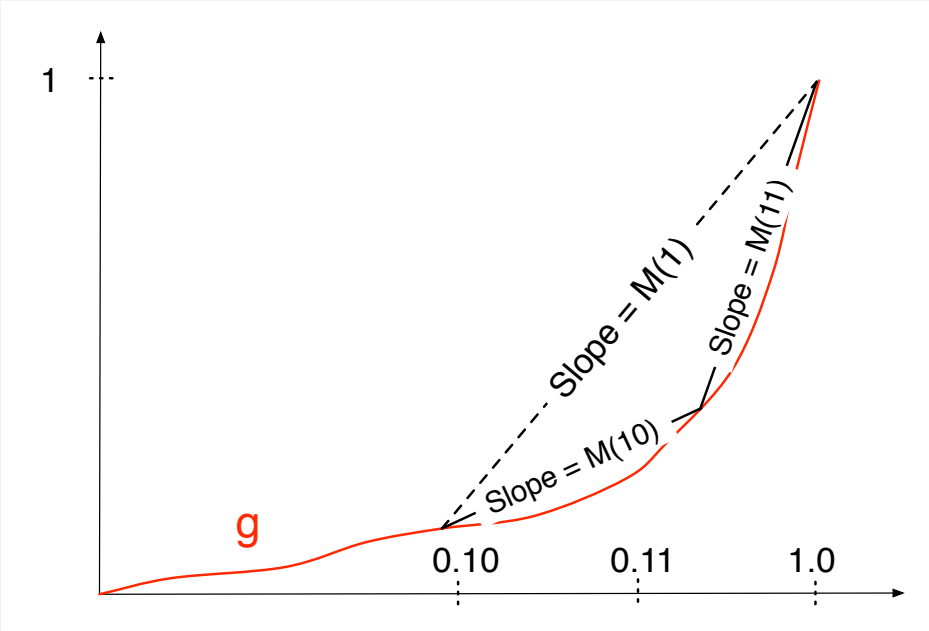
- This also yields our new proof of Demuth's result that z is ML-random $\Rightarrow$ each computable function of bounded variation is differentiable at z.
- We use that each function of bounded variation is the difference of two nondecreasing functions.

Nondecreasing functions vs. betting strategies

Proof that r computably random $\Rightarrow g'(r)$ exists.

We prove the contraposition. In the simplest case, suppose that the lower derivative $\underline{D}g(r)$ equals ∞. Then the following computable betting strategy M succeeds on r: for a binary string σ, $M(\sigma)$ is the slope of g between the points $0.\sigma$ and $0.\sigma + 2^{-|\sigma|}$.

This is clearly a betting strategy:
the picture shows,
for instance, that
$2M(1) = M(10) + M(11)$.



This concludes Part I.

Instead of a break...

To see where the English word randomness comes from, we open

THE

BYRTH, LYF, AND ACTES

OF

KYNG ARTHUR;

OF HIS NOBLE KNYGHTES OF THE ROUNDE TABLE,

THEYR MERVEYLLOUS ENQUESTES AND ADUENTURES,

Thachyeuyng of the Sanc Greal;

AND IN THE END

LE MORTE DARTHUR,

WITH THE DOLOUROUS DETH AND DEPARTYNG OUT OF THYS WORLDE OF THEM AL.

WITH AN

INTRODUCTION AND NOTES,

BY ROBERT SOUTHEY, ESQ.

VOL. I.

Etymology of "randomness"

Sir Thomas Malory, Le Morte D'Arthur, Book I.10 (1485):

they were messagers vnto kyng Ban & Bors sent from kynge Arthur, therfor said the viii knyghtes ye shalle dye or be prysoners, for we ben knyghtes of kyng Claudas. And therwith two of them dressid their sperys, and Ulfyus and Brastias dressid theire speres, and ranne to gyder with grete raundon. And Claudas knyghtes brack their speres, and ther to hylde and bare the two knyghtes out of her sadels to the erthe, and so lefte hem lyeng and rode her wayes. And the other sixe knyghtes rode afore to a passage to mete wyth hem ageyne, and so Ulfyus & Brastias smote other two doun And so past on her wayes. And at the fourth passage there mette two for

The old French noun "raundon" is derived from randir, "to gallop". It has been used in English since the 14th Century.

Randomness in various languages

French	chance	
German	Zufall	
Russian	sluchainost	
Polish	losowość	(‘Los’ means ‘fate’)
Georgian	shemtkhvevitoba	
Spanish	aleatoriedad	(Latin ‘alea’ means ‘dice’)
Hebrew	mikriyut	(‘mikre’ - occurrence)
Greek	$\tau\upsilon\chi\alpha\iota\sigma\sigma$ (tuchaios)	(identical in ancient Greek)
Mandarin	随机 $(Su\acute{i}\ j\bar{i})$.	

Part II

Studying computational lowness via randomness

Basics of computability theory 2

A function $\psi\colon\ \mathbb{N} \to \mathbb{N}$ is partial computable if there is a Turing program which, with n on the input tape, outputs $\psi(n)$ if defined, and loops forever otherwise.

if $\psi(n)$ is defined

☹ if $\psi(n)$ is undefined

We say that $A \subseteq \mathbb{N}$ is computably enumerable (c.e.) if A is the domain of a partial computable function. Equivalently, one can effectively enumerate the elements of A in some order.

Basics of computability theory 3

$(W_e)_{e\in\mathbb{N}}$ is an effective listing of all the computably enumerable sets.

The halting problem is a universal computably enumerable set:

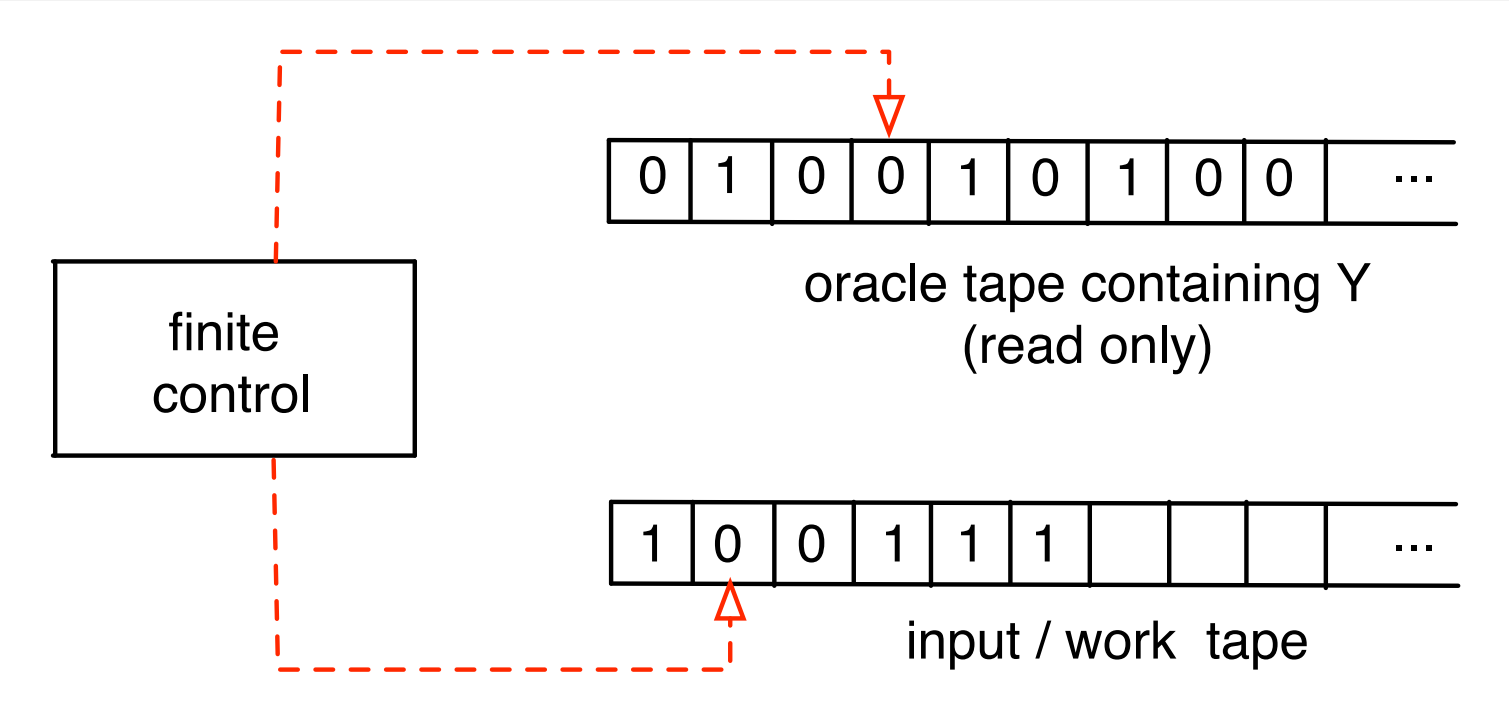
$$\mathcal{H} = \{\langle x, e\rangle : \ x \in W_e\}.$$

Basics of computability theory 4

For sets $X, Y \subseteq \mathbb{N}$, we write

$$X \leq_T Y$$

(X is Turing below Y) if an "oracle" Turing machine can compute X asking queries to Y on its oracle tape.



2 Studying lowness via randomness

- More background on algorithmic randomness
- K-triviality
- Existence outside the computable sets, and the cost of changes
- Coincidence with lowness for ML-randomness

Descriptive string complexity K

Consider a partial computable function from binary strings to binary strings. It is called prefix-free if its domain is an antichain under the prefix relation of strings.

There is a universal prefix-free partial computable function $\mathbb{U}$: for every prefix-free function M,

$$M(\sigma) = y \text{ implies } \exists \tau\, \mathbb{U}(\tau) = y, \text{ where } |\tau| \leq |\sigma| + d_M,$$

and the constant d_M only depends on M.

The prefix-free Kolmogorov complexity is the length of a shortest $\mathbb{U}$-description of y:

$$K(y) = \min\{|\sigma| : \mathbb{U}(\sigma) = y\}.$$

Degree of randomness for sequences of bits

00000000 00000000 00000000 00000000 0000...

10100100 01000010 00001000 00010000 0001...

00100100 00111111 01101010 10001000 1000 ...

10010100 00010001 11110100 00101101 1111 ...

11101101 01111010 10101111 11001110 1110 ...

Explanations:

Only zeros

$\prod_i 0^i 1$

$\pi - 3$ in binary

Coin tossing

Coin tossing

Schnorr's 1973 Theorem

We think of a string τ as random if it is incompressible:
$K(\tau) > |\tau| - b$ for some small constant b.
For an infinite sequence of bits Z let

$$Z\upharpoonright_n = Z(0)\ldots Z(n-1).$$

An infinite sequence of bits Z is Martin-Löf random iff each of its initial segments is random as a string:

Theorem (Schnorr 1973; Levin)

Z is ML-random $\iff$ there is $b \in \mathbb{N}$ such that $\forall n\,[K(Z\upharpoonright_n) > n - b]$.

An example of a ML-random real is Chaitin's halting probability

$$\Omega = \sum_{\mathbb{U}(\sigma)\downarrow} 2^{-|\sigma|}.$$

In particular, Ω is not computable.

2 Studying lowness via randomness

- More background on algorithmic randomness
- K-triviality
- Existence outside the computable sets, and the cost of changes
- Coincidence with lowness for ML-randomness

Definition of K-triviality

In the following we identify a natural number n with the string that is its binary representation. For a string τ, up to additive constants we have $K(|\tau|) \leq K(\tau)$, since we can compute $|\tau|$ from τ.

Definition (Chaitin, 1975)

A sequence of bits A is K-trivial if, for some $b \in \mathbb{N}$,

$$\forall n \, [K(A\upharpoonright_n) \leq K(n) + b],$$

namely, all its initial segments have minimal K-complexity.

It is not hard to see that $K(n) \leq 2 \log_2 n + O(1)$.

$$\begin{array}{llll} Z \text{ is} & \text{ML-random} & \Leftrightarrow & \forall n \, [K(Z\upharpoonright_n) > n \quad -O(1)] \\ A \text{ is} & K\text{-trivial} & \Leftrightarrow & \forall n \, [K(A\upharpoonright_n) \leq K(n) \quad +O(1)] \end{array}$$

Thus, being K-trivial means being far from random.

Properties of the K-trivials

- Chaitin (1975) proved that for each constant b there are only $O(2^b)$ K-trivials. This implies that each K-trivial set is Turing below the halting problem $\mathcal{H}$.
- Solovay (1976) built a non-computable K-trivial set.
- This was improved to a computably enumerable example by Downey, Hirschfeldt, Nies, and Stephan (2002).
- They also showed that no K-trivial is Turing equivalent to the halting problem $\mathcal{H}$.

2 Studying lowness via randomness

- More background on algorithmic randomness
- K-triviality
- Existence outside the computable sets, and the cost of changes
- Coincidence with lowness for ML-randomness

Building a non-computable c.e. K-trivial set A

Let $K_s(w) = \min\{|\sigma| : \mathbb{U}(\sigma) = w \text{ in } s \text{ steps}\}$.

To make A non-computable, we may have to put a number x of our choice into A in order to diagonalize against the e-th computable function that claims to be the characteristic function of A.

In this case, for any $w \in \mathbb{N}$, $w > x$, we have to provide short descriptions, of length $K_s(w) + O(1)$, of the new $A\upharpoonright_w$.

If for each e, we can choose the number x so large that

$$\sum_{w=x+1}^{s} 2^{-K_s(w)} \leq 2^{-e-2},$$

then we will never run out of measure for new descriptions in the prefix-free machine we are building, because $\sum_e 2^{-e-2} = 1/2$.

The Limit Lemma

To study restrictions on the K-trivials, we need the following:

Theorem (Shoenfield Limit Lemma,1959)

A is Turing below the halting problem $\mathcal{H}$ $\iff$

there is a computable function $g\colon \mathbb{N}\times\mathbb{N}\to\{0,1\}$ such that

$$A(x) = \lim_s g(x,s)$$

for each $x\in\mathbb{N}$.

We will write $A_s(x)$ for $g(x,s)$.

K-trivials and the total cost of changes 1

Recall Chaitin's result that each K-trivial set A is Turing below the halting problem.
We will show that A has a computable approximation with a small total of changes. The cost of changing $A(x)$ at stage s is

$$c(x,s) = \sum_{w=x+1}^{s} 2^{-K_s(w)}.$$

This is the function we have already used in the existence proof. It is computable.

K-trivials and the total cost of changes 2

$$\begin{aligned} K_s(w) &= \min\{|\sigma| : \mathbb{U}(\sigma) = w \text{ in } s \text{ steps}\} \\ c(x,s) &= \sum_{w=x+1}^{s} 2^{-K_s(w)} \end{aligned}$$

Theorem (N, 2005)

A is K-trivial $\iff$ A has a computable approximation $(A_s)_{s\in\mathbb{N}}$ such that $\infty > \sum_{x,s}\{c(x,s) :\ x < s \ \wedge\ x$ is least s.t. $A_{s-1}(x) \neq A_s(x)\}$.

- The implication "$\Leftarrow$" is by the same argument as in the existence proof for non-computable K-trivials: we can build a prefix-free machine that keeps up with the changes of A.
- The implication "$\Rightarrow$" is hard. It needs the "golden run" method.

K-trivials and the total cost of changes 3

Corollary

Each K-trivial set A is Turing below a computably enumerable K-trivial set D.

- D is the "change set" $\{\langle x, i\rangle : A_s(x)$ changes at least i times$\}$.
- Thus, whenever $A_s(x) \neq A_{s-1}(x)$, we put the next element in the "column" x into D.
- Changing D is cheaper than changing A, so the total cost of changes for D is finite.

2 Studying lowness via randomness

- More background on algorithmic randomness
- K-triviality
- Existence outside the computable sets, and the cost of changes
- Coincidence with lowness for ML-randomness

Lowness for Martin-Löf randomness

The following specifies a sense in which a set A is computationally weak when used as an oracle.

Definition

A is low for Martin-Löf randomness if every ML-random set Z is already ML-random with oracle A.

- This property was introduced by Zambella (1990).
- Kučera and Terwijn (1999) built a c.e. non-computable set of this kind.
- Low for computably random $\Rightarrow$ computable (Nies, 2005).

Far from random = close to computable

Theorem (2005)

Let $A \subseteq \mathbb{N}$. Then

$$A \text{ is } K\text{-trivial} \iff A \text{ is low for Martin-Löf randomness.}$$

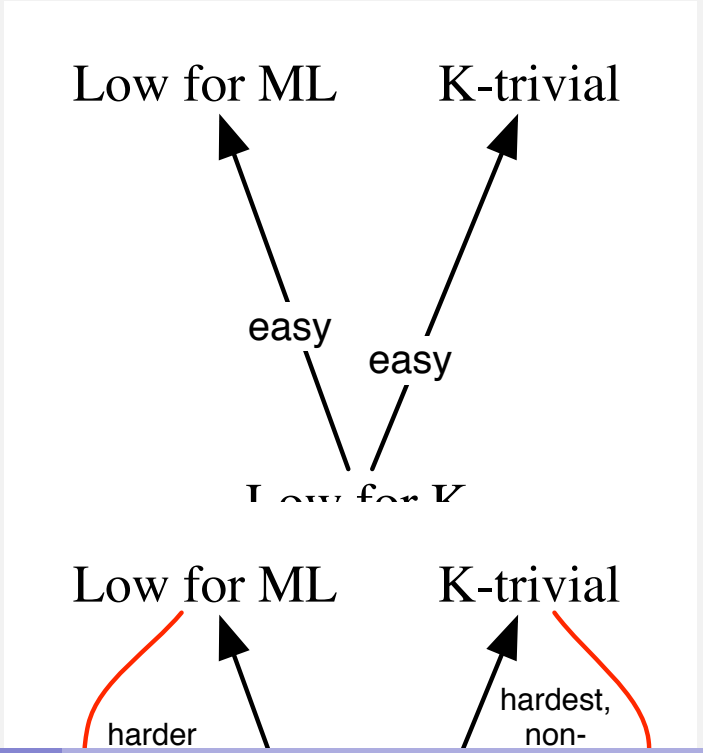
- The implication "$\Rightarrow$" is joint with Hirschfeldt.
- It improved the result Nies had obtained previously that the K-trivials are closed downward under $\leq_{\mathrm{T}}$.
- As a consequence of the downward closure under $\leq_T$, the K-trivials induce a Σ^0_3 ideal in the Turing degrees below the halting problem.
- Furthermore, the K-trivial real numbers form a real closed field.

Lowness for K

A is called low for K (Muchnik, 1998) if enhancing the computational power of the universal function $\mathbb{U}$ by an oracle A does not decrease $K(y)$:

$$\forall y\,[K(y) \leq K^A(y) + O(1)].$$

- The straightforward implications are
  low for $K \Rightarrow$ low for ML and
  low for $K \Rightarrow$ K-trivial.
- In N (2005) we show the converse implications.



Lowness paradigms

We have seen two paradigms for lowness of a set A:

- Inertness: A can be computably approximated with small total of changes (e.g. K-triviality).
- Oracle weakness: A is not very useful as an oracle (e.g., lowness for Martin-Löf randomness).

Further lowness paradigm:

- It is easy for oracles to compute A. In some sense, "many oracles" compute A.

An instance of the "easy-to-compute" paradigm

A is incomplete ML-coverable (Hirschfeldt, N, Stephan 2004) if $A \leq_{\mathrm{T}} Y$ for some ML-random $Y \not\geq_T \mathcal{H}$.

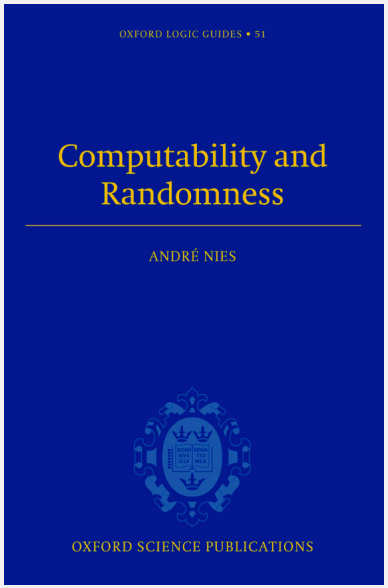
- For c.e. sets A, incomplete ML-coverable $\Rightarrow$ K-trivial.
- A main open question in the area is whether the converse implication holds.

Characterizing a class via the "easy-to-compute" paradigm

Let A be computably enumerable. We say that A is strongly jump-traceable (Figueira, Nies, Stephan) if there are very few choices for $\psi(x)$, for any partial computable in A function ψ.

- This is a definition according to the "oracle weakness" paradigm.
- However, Greenberg, Hirschfeldt and N have characterized it via the "easy-to-compute" paradigm:
- strong jump traceability is equivalent to being below each ω-c.e. ML-random set Y.
  (We say that Y is ω-c.e. if $Y(n)$ can be computably approximated with a computably bounded number of changes.)

References



My book "Computability and Randomness",
Oxford University Press, 447 pages, Feb. 2009;

The proceedings paper; these and other slides, on my web page.