Applying randomness to computability

André Nies
The University of Auckland

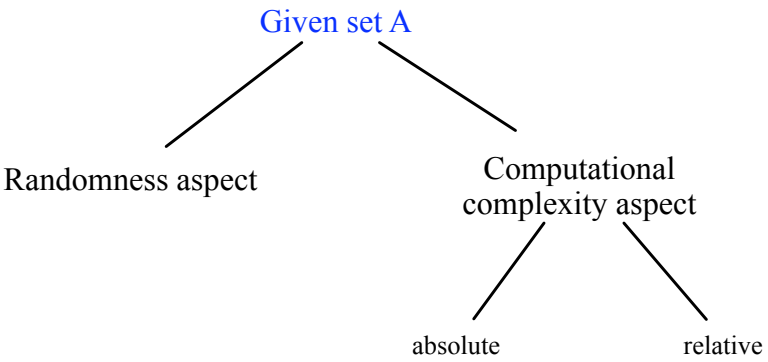
ASL meeting, Sofia, 2009

Thanks to:

- Alexandra, Mariya and the rest of the crew
- Andrej Bauer, Laurent Bienvenu, George Barmpalias, Mathieu Hoyrop, Santiago Figueira for proofreading the slides.

Two aspects of a set of natural numbers

- We study sets of natural numbers. We refer to them simply as **sets**.
- We consider two aspects of a set, the **randomness** and the **computational complexity** aspect.



Principal Thesis

The two aspects of sets,

- randomness
- computational complexity,

are very closely related.

A little history

- Up to the year 1998 or so, the randomness aspect received little attention from computability theorists.
- It was confined to very good but isolated work such as Solovay's manuscript (1975), Kurtz' thesis (1981), Kautz' thesis (1990), little-known work by Demuth, and Kučera's 1985-1993 papers.
- More recently, it has become widely known that randomness-related concepts enrich computability theory.
- For instance, they help us to understand properties saying a set is very close to being computable.

Important work 1999-2003

- '**Lowness for the class of random sets**" (1999) by Kučera and Terwijn;
- "**R.e. reals and Chaitin Omega numbers**" by Calude, Hertling, Khoussainov, Wang (2001); subsequent paper "**Randomness and recursive enumerability**" by Kučera and Slaman (2001);
- "**Trivial reals**" by Downey, Hirschfeldt, Nies, Stephan (2003).
- "**Lowness properties and randomness**" by Nies (published 2005).

The plan for Lecture 1

We provide background.

- We review some important concepts from computability. We clarify the computational complexity aspect of a set.
- We define descriptive complexity of strings x, introducing the complexity measure $K(x)$.

Also today, we consider the classical interaction,
from computability **to** randomness.

- We use algorithmic tools to introduce a hierarchy of mathematical randomness notions.
- We study these notions, again with algorithmic tools.

The interaction from randomness to computability

- In the Lectures 2 and 3, we apply randomness to understand the computational complexity of sets.
- More generally, we show how randomness-related concepts enrich computability theory.

Lecture 3 also covers some connections to effective descriptive set theory.

Some references

Calibrating randomness by Downey, Hirschfeldt, N, Terwijn. **BSL** 12 (2006), 411-491.
My book "**Computability and Randomness**",
Oxford University Press, 447pp, Feb. 2009.

Forthcoming book "**Algorithmic randomness and complexity**" by Downey and Hirschfeldt.

These slides.

Further material:

Recent talk "**New directions in computability and randomness**" (open questions); recent workshop tutorials "**Cost functions**", "**Randomness via effective descriptive set theory**" all on my web site.

Part I

Computability and its application to randomness

1 Background from computability theory

Oracle computations

- All concepts in computability are ultimately based on (Turing machine) computations.

- Equipping the machine with an oracle X, we get a version of the concept relative to X.

- For instance, "A is computable" becomes "A is computable in X", written $A \leq_T X$.

Turing functionals

- Let $(M_e)_{e\in\mathbb{N}}$ be an effective listing of all oracle Turing machines.
  Let $(\Phi_e)_{e\in\mathbb{N}}$ be the corresponding effective list of Turing functionals.
  Φ_e^X is the partial function determined by M_e with oracle X.

- Thus, $A \leq_T X \iff \exists e\,[A = \Phi_e^X]$. (Here we identify the set $A \subseteq \mathbb{N}$ with its characteristic function $\mathbb{N} \to \{0,1\}$.)

- $A \leq_{tt} X$ if in addition, the running time is computably bounded in the input. (This is called truth-table reducibility because the oracle computations can be viewed as evaluating an effectively determined Boolean function on the answers to oracle queries.)

The Turing jump

- $J^X(e)$ is a universal Turing functional.
  For instance, let $J^X(e) \simeq \Phi_e^X(e)$.

- $X' = \{e\colon\ J^X(e) \text{ is defined}\}$ is the **halting problem relative to** X.

- X' is computably enumerable relative to X, and $X' >_T X$.

- $\emptyset'$ is the usual **halting problem**.

- $A \leq_T B$ implies $A' \leq_T B'$.

- A is $\Delta^0_2 \iff A \leq_T \emptyset'$.

The Shoenfield Limit Lemma

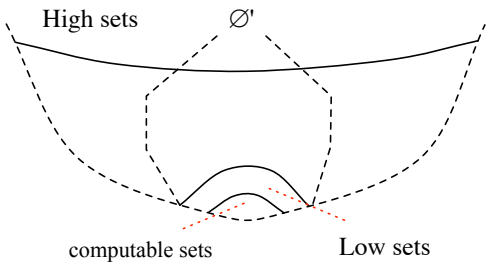
- A is $\Delta^0_2 \iff A(x) = \lim_s f(x,s)$ for some computable $\{0,1\}$-valued function f. Often, we don't name such an approximating function explicitly, and rather write $A_s(x)$ instead of $f(x,s)$.

- $A \leq_{tt} \emptyset'$ if in addition, the number of changes in some computable approximation is computably bounded in x. (Such a set is called **ω-c.e.**)

Lowness properties and highness properties

- A **lowness property** specifies a sense in which a set is close to computable. It is closed **downwards** under $\leq_T$.
- A **highness property** specifies a sense in which a set almost computes $\emptyset'$. It is closed **upwards** under $\leq_T$.

Example 1. $A \subseteq \mathbb{N}$ is **low** if $A' \leq_T \emptyset'$, and A is **high** if $\emptyset'' \leq_T A'$.



Lowness, highness, domination

By the Limit Lemma, $A' \leq_T \emptyset' \iff$ there is a computable 0,1-valued approximation $f(x,s)$ to the question whether $x \in A'$. That is,

$$A'(x) = \lim_s f(x,s).$$

Highness $\emptyset'' \leq_T A'$ can be characterized via domination.

Theorem 2 (Martin 1966)**.** *A is* **high** $\iff$ *some function $f \leq_T A$ dominates each computable function (on almost all arguments).*

A set can be complex in one sense, and computationally weak in another sense. For instance, there is a high set of minimal Turing degree (Cooper).

Conventions

- x, y, z, σ, τ denote strings from $\{0,1\}^*$.
- We can identify strings with numbers via a computable bijection $\{0,1\}^* \leftrightarrow \mathbb{N}$ (related to the binary presentation).
- The capital letters A, B, C, X, Y, Z denote sets of natural numbers.

Cantor Space

Sets are identified with infinite sequences of 0,1. They form the **Cantor space** $2^{\mathbb{N}}$, which is equipped with the product topology.

- $[\sigma] = \{X : \sigma \prec X\}$ is the class of sets extending the string σ.
  The $[\sigma]$ form a basis of clopen classes for the topology.
- Thus, an **open class** (or set) has the form $\bigcup_{\sigma \in C} [\sigma]$.
- Such a class is **computably enumerable**, or Σ^0_1, if one can choose C computably enumerable.
- The complements of c.e. open classes are called Π^0_1 **classes**.
- A Π^0_1 class is given as the set of paths through a computable binary tree.
- It is easy to construct a non-empty Π^0_1 class without a computable member.

Basis Theorems for Π^0_1 classes

Definition 3. A is computably dominated $\iff$ each function $f \leq_T A$ is dominated by a computable function.

This lowness property is incompatible with being low in the usual sense. In fact, the only computably dominated Δ^0_2 sets are the computable sets.
A **basis theorem** (for Π^0_1 classes) says that each non-empty Π^0_1 class has a member with a particular property (often, a lowness property).

Theorem 4 (Jockusch and Soare 1972)**.** *Let $\mathcal{P}$ be a non-empty Π^0_1 class.*

- *$\mathcal{P}$ has a low member.*
- *$\mathcal{P}$ has a computably dominated member.*

2 Descriptive string complexity

Prefix free machines

- Let $\{0,1\}^*$ be the strings over $\{0,1\}$. A **machine** is a partial recursive function $M : \{0,1\}^* \mapsto \{0,1\}^*$.
- If $M(\sigma) = x$ we say that σ is an **M-description** of x.
- We say that M is **prefix free** if no M-description is a proper initial segment of any other M-description.

The universal prefix free machine, and $K(x)$

- Let $(M_d)_{d\in\mathbb{N}}$ be an effective listing of all prefix free machines. We define a **universal** prefix free machine $\mathbb{U}$ by
$$\mathbb{U}(0^d 1\sigma) \simeq M_d(\sigma).$$
- Given string x, the **prefix free descriptive string complexity** $K(x)$ is the length of a shortest $\mathbb{U}$-description of x:
$$K(x) = \min\{|\sigma| : \mathbb{U}(\sigma) = x\}.$$

Some facts about K
Let "$\leq^+$ denote "$\leq$" up to a constant. (For instance, $2n + 5 \leq^+ n^2$.)
Let $|x| \in \{0,1\}^*$ denote the length of a string x, written in binary.
The following bounds are proved by constructing appropriate prefix free machines.

- For each computable function f we have
$$K(f(x)) \leq^+ K(x).$$
In particular, we have the **lower bound** $K(|x|) \leq^+ K(x)$.
- We have the **upper bound**
$$K(x) \leq^+ |x| + K(|x|).$$
- Also $K(|x|) \leq^+ 2\log|x|$, so the upper bound is not far beyond $|x|$.
- If $K(x) \geq^+ |x|$ we think of x as **incompressible**. This formalizes the intuitive notion of randomness for strings.

Quiz 1

(1) Can a high set be computably dominated?

(2) Is "not low", i.e $A' \not\leq_T \emptyset'$, a highness property?

(3) Define a prefix free machine showing that $K(x) \leq^+ |x| + K(|x|)$.

3 Randomness for infinite objects

Reminder: computational complexity

We have given some formal treatment of the computational complexity aspect of a set A.

absolute complexity: the membership of A in classes of similar complexity, such as lowness, or highness;

relative complexity: comparing A to other sets via reducibilities such as $\leq_T$ or $\leq_{tt}$.

Can we do the same for the randomness aspect?

Measure

- The **product measure** λ on Cantor space $2^{\mathbb{N}}$ is given by
$$\lambda[\sigma] = 2^{-|\sigma|}$$
for each string σ.
- If $G \subseteq 2^{\mathbb{N}}$ is open then $\lambda G = \sum_{\sigma \in B} 2^{-|\sigma|}$ where B is a prefix free set of strings such that $G = \bigcup_{\sigma \in B}[\sigma]$.
- $\mathcal{C} \subseteq 2^{\mathbb{N}}$ is **null** if $\mathcal{C}$ is contained in some Borel $\mathcal{B}$ such that $\lambda \mathcal{B} = 0$.

Null classes and randomness

The intuition: an object is random if it satisfies no exceptional properties.

Here are some examples of exceptional properties of a set Y:

- Having every other bit zero:
$$\forall i \, Y(2i) = 0.$$
- Having at least twice as many zeros as ones in the limit:
$$2/3 \leq \liminf |\{i < n : Y(i) = 0\}|/n.$$

We would like to formalize "exceptional property" by "null class".

- The examples above are null classes, so they should not contain a random set.
- The problem: if we do this, no set Z is random, because $\{Z\}$ itself is a null class!

The solution: effective null classes

- Using algorithmic tools, we introduce **effective null classes**, also called **tests**.
- To be random in an algorithmic sense, Z merely has to avoid these effective null classes, that is, pass those tests.

Tests

Fact 5. *The class $\mathcal{C} \subseteq 2^{\mathbb{N}}$ is null $\iff$ $\mathcal{C} \subseteq \bigcap G_m$ for some sequence $(G_m)_{m \in \mathbb{N}}$ of open sets such that $\lambda G_m \to 0$.*

We obtain a type of effective null class (or test) by adding **effectivity restrictions** to this condition characterizing null classes.

(1) Require an **effective presentation** of $(G_m)_{m \in \mathbb{N}}$;

(2) possibly, require **fast** convergence of $\lambda G_m \to 0$.

By (1) there are only countably many effective null classes. Their union is still a null class and contains all non-random sets in this sense. So each effective randomness notion has measure 1.

Martin-Löf's randomness (1966)

Definition 6.

- A **Martin-Löf test** is a uniformly computably enumerable sequence
$$(G_m)_{m\in\mathbb{N}}$$
of open sets in $2^{\mathbb{N}}$ such that
$$\lambda G_m \leq 2^{-m} \text{ for each } m.$$
- A set Z is **Martin-Löf random** if Z passes each ML-test, in the sense that Z is not in all of the G_m.

The previous examples (every other bit is zero/ in the limit at least twice as many zeros as ones) are effective null classes in this sense. So a ML-random set cannot have these properties.

Chaitin's halting probability

We identify coinfinite sets with real numbers in $[0, 1)$.

- Consider the halting probability of the universal prefix free machine $\mathbb{U}$,
$$\Omega = \sum_{\mathbb{U}(\sigma)\text{ halts}} 2^{-|\sigma|}.$$
- Note that this sum converges because $\mathbb{U}$ is prefix free.
- Ω is Martin-Löf random (Chaitin).
- The left cut $\{q \in \mathbb{Q}\colon\ q < \Omega\}$ is computably enumerable.
- $\Omega \equiv_T \emptyset'$.

Characterization via the initial segment complexity

Given Z and $n \in \mathbb{N}$, let $Z\upharpoonright_n$ denote the initial segment $Z(0)\dots Z(n-1)$.

Schnorr's Theorem says that Z is ML-random $\iff$ each initial segment is incompressible.

Theorem 7 (Schnorr 1973)**.** *Z is ML-random $\iff$ there is $b \in \mathbb{N}$ such that $\forall n\, K(Z\upharpoonright_n) \geq n - b$.*

Levin (1973) proved the analogous theorem for monotone string complexity.

A universal Martin-Löf test

Schnorr's Theorem yields a **universal** ML-test:

- Let $\mathcal{R}_b = \{X\colon\ \exists n\,[K(X\upharpoonright_n) < n - b]\}$.
- $\mathcal{R}_b$ is uniformly c.e. open, because the relation "$K(x) < r$" is Σ^0_1.
- One shows that $\lambda\mathcal{R}_b \leq 2^{-b}$.
- With this notation, Schnorr's Theorem says that
Z is ML-random $\iff$ Z passes the test $(\mathcal{R}_b)_{b\in\mathbb{N}}$.
- So, the single test $(\mathcal{R}_b)_{b\in\mathbb{N}}$ is enough!

ML-random sets with lowness properties

- Consider the complement of $\mathcal{R}_1$:

$$\{X\colon\ \forall n\, K(X\upharpoonright_n) \geq n-1\}.$$

  This is a Π^0_1 class of measure $\geq 1/2$.
- By Schnorr's Theorem, it consists entirely of ML-random sets.
- So we can apply the Jockusch-Soare basis theorems to obtain random sets with lowness properties:

Example 8.

- There is a low ML-random set.
- There is a computably dominated ML-random set.

Schnorr randomness

Schnorr critized that Martin-Löf-randomness is too strong to be considered algorithmic. He proposed a restricted test notion where we know more about the tests.

- A **Schnorr test** is a ML-test $(G_m)_{m\in\mathbb{N}}$ such that λG_m is a computable real uniformly in m.
- Z is **Schnorr random** if $Z \notin \bigcap_m G_m$ for each Schnorr test $(G_m)_{m\in\mathbb{N}}$.

It suffices to consider the tests $(G_m)_{m\in\mathbb{N}}$ where $\lambda G_m = 2^{-m}$.

Schnorr randoms in each high degree

Theorem 9 (Nies, Stephan, Terwijn, 2005; Downey, Griffiths, 2005)**.** *For each high set A there is a Schnorr random set $Z \equiv_T A$.*

- Schnorr random sets satisfy all statistical criteria for randomness, such as the law of large numbers.
- They do **not** satisfy all computability-theoretic criteria.
- For instance, there is a high minimal Turing degree, hence a Schnorr random set Z of minimal degree.
  - If Z_0 is the bits in even position, and Z_1 the bits in odd position, then both Z_0, Z_1 are incomputable.
  - Hence $Z_0 \equiv_T Z_1$.
  - Note that such a set Z is not ML-random.

Theory of Schnorr randomness

- The **theory** of Schnorr randomness parallels the theory of ML-randomness.
- This is surprising because there is no universal Schnorr test.
- We will see more parallels when we get to the lowness notions.
- The actual **results** are quite different.

Characterization via the initial segment complexity

A prefix free machine M is called **computable measure machine** if its halting probability $\Omega_M = \sum_{M(\sigma)\,\text{halts}} 2^{-|\sigma|}$ is a computable real number. For instance, define M by $M(0^n1) = n$ in binary. Then $\Omega_M = 1$.

- Recall Z is Martin-Löf random $\iff$ for some b, $\forall n\ K(Z\upharpoonright_n) \geq n - b$.
- The following is an analog for Schnorr randomness. Since there is no universal computable measure machine, we have to quantify over all of them.

- For a prefix free machine M,

$$K_M(x) = \text{length of a shortest } M\text{-description of } x.$$

Theorem 10 (Downey, Griffiths, 2005). *Z is Schnorr random $\iff$ for each computable measure machine M, for some b, $\forall n\ K_M(Z\upharpoonright_n) \geq n - b$.*

Weak 2-randomness

One can also argue that ML-randomness is too **weak**.
For instance, the ML-random real Ω has a computably enumerable left cut, and computes the halting problem. These properties may contradict our intuition of randomness.
Recall: a ML-test is a uniformly computably enumerable sequence $(G_m)_{m\in\mathbb{N}}$ of open sets such that $\lambda G_m \leq 2^{-m}$ for each m.

- If the condition "$\lambda G_m \leq 2^{-m}$ for each m" is removed (and we only retain the basic condition $\lim_m \lambda G_m = 0$), we obtain a test concept equivalent to null Π^0_2 classes.
- We say that Z is **weakly 2-random** if Z is in no null Π^0_2 class.

If Z is Δ^0_2 then $\{Z\}$ is a Π^0_2 class. Thus, no Δ^0_2 set is weakly 2-random. This rules out Ω as weakly 2-random.

2-randomness

Finally, Z is called **2-random** if Z is ML-random relative to $\emptyset'$.
We have the proper implications

$$\text{2-random} \Rightarrow \text{weakly 2-random} \Rightarrow \text{Martin-Löf} \Rightarrow \text{Schnorr}.$$

2-randomness can be characterized by strong incompressibility of *infinitely many* initial segments.
Let $C(x)$ denote the plain Kolmogorov complexity of a string x. Note that $C(x) \leq^+ |x|$.

Theorem 11. *Z is 2-random*
$\iff$ $C(Z\upharpoonright_n) =^+ n$ for infinitely many n (N,S,T, 2005; Miller, 2005)
$\iff$ $K(Z\upharpoonright_n) =^+ n + K(n)$ for infinitely many n (Miller, 2008).

No initial segment characterization is known for weak 2-randomness.

4 Studying randomness notions via computability

Randomness enhancement via lowness properties

A randomness enhancement result has the form

Weaker randomness notion & **lowness property** $\Rightarrow$ stronger randomness notion.

Intuitively speaking, if Z is random in the weaker sense, then being computationally **less** complex implies being **more** random.

Enhancing Schnorr randomness

Theorem 12 (Nies, Stephan, Terwijn 2005). *Let Z be Schnorr random and not high. Then Z is already ML-random.*

Theorem 13 (Nies, Stephan, Terwijn 2005; Yu Liang). *Let Z be Schnorr random and computably dominated. Then Z is already weakly 2-random.*

Proof. Suppose $Z \in \bigcap_m G_m$ where $(G_m)_{m\in\mathbb{N}}$ is uniformly c.e. and $\lambda G_m \to 0$. Let $G_{m,s}$ be the clopen set approximating G_m at stage s. Let

$$f(m) \text{ the least } s \text{ such that } Z \in G_{m,s}.$$

Then $f \leq_T Z$, so there is a computable function h dominating f.
Now Z fails the Schnorr test $(G_{m,h(m)})_{m\in\mathbb{N}}$. □

Enhancing Martin-Löf randomness
Here, the randomness notion stronger than ML-randomness actually implies the lowness property. So we have characterization :
Martin-Löf randomness $\Longrightarrow$

lowness property $\iff$ stronger randomness notion.

Theorem 14 (Hirschfeldt, Miller 2006)**.** *Let Z be ML-random. Then every Δ^0_2 set Turing below Z is computable $\iff$ Z is weakly 2-random*

"$\Leftarrow$" extends the fact that a weakly 2-random is not Δ^0_2.
"$\Rightarrow$" relies on the cost function method of Lecture 2. □

We say that Z is **low for** Ω if Chaitin's Ω is ML-random relative to Z.

Theorem 15 (Nies, Stephan, Terwijn 2005)**.** *Let Z be ML-random. Then*
Z is low for Ω $\iff$ Z is 2-random.

This uses a theorem of van Lambalgen on relative ML-randomness.

Summary of Part 1: (1) the computational complexity aspect of sets

- The absolute computational complexity of a set is given by membership in classes, such as low, high, computably dominated.
- Lowness properties don't form a hierarchy; they can exclude each other (outside the computable sets).
- The relative computational complexity is given by reducibilities such as $\leq_T$.

Summary of Part 1: (2) the randomness aspect of sets

- The intuitive notion of randomness of a set corresponds to a hierarchy of mathematical randomness notions.
- The central notion is Martin-Löf-randomness.
- Schnorr and weak 2 randomness can be seen as variants. 2-randomness is the relativization to $\emptyset'$.
- All the notions except weak 2-randomness have been characterized by incompressibility of initial segments.
- By and large, if a set already satisfies a randomness notion, being computationally less complex implies being more random.

Quiz 2

(1) Which one holds? Schnorr's Theorem is about

- Schnorr randomness
- Martin-Löf randomness
- cats.

(2) Give an example of a 2-random set. Give examples showing that the implications are proper:

$$\text{2-random} \Rightarrow \text{weak 2-random} \Rightarrow \text{Martin-Löf} \Rightarrow \text{Schnorr}.$$

Note here that no 2-random set is computably dominated (Kurtz, 1981).

(3) Suppose A is computable. If Z satisfies a randomness notion, show that $Z \Delta A$ satisfies the same notion.

Part II

Lowness properties within the Δ^0_2 sets

The plan for Part 2

- We introduce lowness properties within the Δ^0_2 sets.
- We investigate them using randomness notions.
- We introduce the K-trivial sets which are far from random.
- The cost function method is used to build Δ^0_2 sets with lowness properties.

Lowness paradigms

A lowness property specifies a sense in which a set is close to computable. There are several paradigms:

1. **Weak as an oracle**.
   Examples: lowness $A' \leq_T \emptyset'$; being computably dominated (each function $f \leq_T A$ is dominated by a computable function).
2. **Easy to compute**. The class of oracles computing A is large in some sense.
   We'll see examples shortly, such as sets computed by random oracles.
3. For a Δ^0_2 set: **Approximable with a small total of mind changes**.

5 Subclasses of the low sets, and the amount of injury

How low is the property $A' \leq_T \emptyset'$?

Definition 16. The coinfinite computably enumerable set A is called **simple** if it has a non-empty intersection with each infinite c.e. set.

Such a set is incomputable as its complement is not c.e.

Some existence results for low sets:

(1) The Low Basis Theorem (Jockusch and Soare 1972).

(2) There is a low simple set (1957). This provides a solution to Post's 1944 problem whether some c.e. set is neither computable nor Turing complete.

(3) There are disjoint low c.e. sets A_0, A_1 such that $\emptyset' = A_0 \cup A_1$. (Sacks Splitting Theorem,1962.)

By **(3)**, lowness does not necessarily mean very close to computable.

Superlowness

Definition 17 (Bickford and Mills 1982; Mohrherr 1986)**.** A is called **superlow** if $A' \leq_{tt} \emptyset'$. That is, one can approximate $A'(x)$ with a computably bounded number of mind changes.

- The proofs of **(1)** and **(2)** actually yield superlowness.
- In contrast, **(3)** fails for superlowness.

Building a low simple set A

To make A **simple** we meet the **simplicity requirements**

$$S_e : \ |W_e| = \infty \Rightarrow A \cap W_e \neq \emptyset.$$

To make A **low**, we meet the lowness requirements

$$L_i : \ J^A(i) \text{ is defined at infinitely many stages} \Rightarrow J^A(i) \text{ converges.}$$

Let $W_e[s]$ be the finite set of numbers that have entered W_e by stage s.

Constraint construction I

At stage s, if S_e is not satisfied yet,

look for an x, $2e \leq x < s$, such that $x \in W_e[s]$ and

Constraint: for each $i \leq e$, if the computation $J^A(i)$ is defined, then $x > q$ for each query q to the oracle A asked in that computation.

If so, put x into A. (This satisfies S_e.)

Injury

- **Injury** to a lowness requirement L_i at a stage s means: $J^A(i)$ is defined at s, and $A(q)$ changes, for an oracle query q in that computation.
- The construction for building a low simple set, and Sacks Splitting, have injury to lowness requirements.
- In the construction of a low simple set, injury to L_i is caused by requirements S_e, $e < i$, enumerating numbers into A.

The amount of injury

- To show lowness, at stage s we guess that
$$i \in A' \text{ if } J^A(i) \text{ is defined at this stage.}$$
- The priority ordering of requirements is L_0, S_0, L_1, S_1, ...
- Since S_e acts at most once, L_i can be injured at most i times. Hence the guess goes wrong at most i times, and A is in fact superlow.
- In Sacks Splitting, the number of injuries to lowness requirements is not computably bounded. In fact we know that the sets cannot both be superlow.
- Our intuition: when we build a c.e. set, the less injury we have, the closer to computable will the set be. The next theorem gives further evidence for this principle:
- we use ML-randomness to provide a solution to Post's problem **without** injury. We will see later on that the resulting set A is **much** closer to being computable than the previous examples.

Kučera's 1985 Theorem (in fact, a special case)

Theorem 18. *Let Y be Δ^0_2 and* **ML-random**. *Then there is a simple set $A \leq_T Y$.*

Now let Y be the bits of Ω in the even positions. Then an easy direct proof shows that Y is low and ML-random (N,S, T 2005). Therefore:

Corollary 1. *We can build without injury a low simple set A.*

Proof of the theorem: A **Solovay test** $\mathcal{G}$ is given by an effective enumeration of strings $\sigma_0, \sigma_1, \ldots$, such that

$$\sum_i 2^{-|\sigma_i|} < \infty.$$

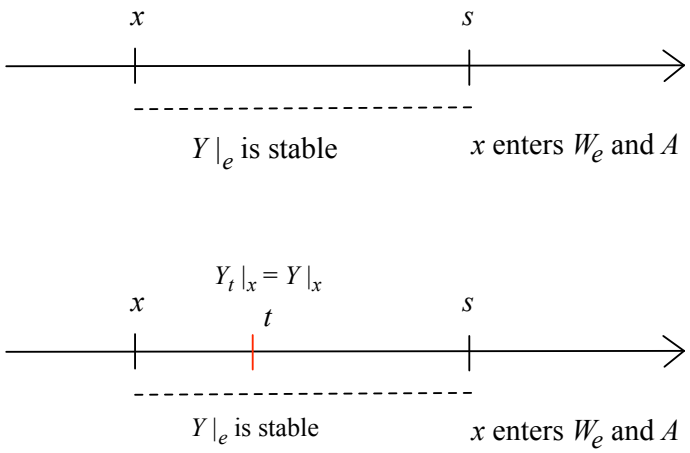
If Y is ML-random, then for almost all i, $\sigma_i \not\preceq Y$.
As before, we meet the simplicity requirements

$$S_e : \ |W_e| = \infty \Rightarrow A \cap W_e \neq \emptyset.$$

Constraint construction II
At stage s, if S_e is not satisfied yet,
look for an x, $2e \leq x < s$, such that $x \in W_e[s]$ and

(Constraint) $\forall t\, [x < t < s \rightarrow Y_t \upharpoonright_e = Y_s \upharpoonright_e]$.

If so, put x into A. Put the string $\sigma = Y_s \upharpoonright_e$ into Solovay test $\mathcal{G}$.



- Each S_e is met, so A is simple.
- $\mathcal{G}$ is a Solovay test, as its total is at most $\sum_e 2^{-e}$.
- To see that $A \leq_T Y$, choose s_0 such that $\sigma \not\preceq Y$ for any σ enumerated into $\mathcal{G}$ after stage s_0.
- Given an input $x \geq s_0$, using Y as an oracle, compute $t > x$ such that $Y_t \upharpoonright_x = Y \upharpoonright_x$. Then $x \in A \leftrightarrow x \in A_t$.
- For, if we put x into A at a stage $s > t$ for the sake of S_e, then $x > e$, so we list σ in $\mathcal{G}$ where $\sigma = Y_s \upharpoonright_e = Y \upharpoonright_e$;

  this contradicts the fact that $\sigma \not\preceq Y$.

6 Lowness properties via randomness

K-trivial $\iff$ low for ML-randomness
Recall: $K(x) =$ length of a shortest prefix free description of string x.

- A set A is **K-trivial** if each initial segment of A has prefix free complexity no greater than the one of its length. That is, there is $b \in \mathbb{N}$ such that, for each n (written in binary),

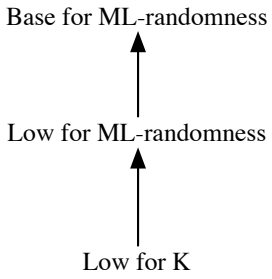
$$K(A\restriction_n) \leq K(n) + b$$

(Chaitin, 1975).

- A is **low for ML-randomness** if each ML-random set is already ML-random relative to A (Zambella, 1990).

We will see that these two concepts are equivalent.
The harder implication is K-trivial $\Rightarrow$ low for ML-randomness. It will be sketched in Lecture 3 (hopefully).

More concepts



Further lowness properties have been introduced:

- **Base for ML-randomness** (Kučera, APAL 1993)
- **Lowness for K** (Muchnik jr., in a Moscow seminar, 1999).

Each time, the existence of a simple set with the property was shown.

Lowness for K

- In general, enhancing the computational power of the universal machine by an oracle A should **decrease** $K(y)$.
- A is **low for K** if this is not so. In other words,

$$\forall y\ K(y) \leq^+ K^A(y).$$

- Exercise in next quiz: low for $K \Rightarrow K$-trivial.

Theorem 19 (Muchnik, 1999)**.** *Some simple set is low for K.*

Fact 20. *If A is low for K then A is generalized low, namely, $A' \leq_T A \oplus \emptyset'$.*

Proof. If $e \in A'_t$, then $K(t) \leq^+ K^A(t) \leq^+ 2\log e$.
$\emptyset'$ can compute $s = \max\{\mathbb{U}(\sigma) : |\sigma| \leq 2\log e + c\}$. Then $e \in A' \iff e \in A'_s$. □

Lowness for ML-randomness
Let MLR denote the class of Martin-Löf-random sets.

- Because an oracle A increases the power of tests, $\mathsf{MLR}^A \subseteq \mathsf{MLR}$.
- A is **low for ML-randomness** if $\mathsf{MLR}^A = \mathsf{MLR}$ (Zambella, 1990).

Theorem 21 (Kucera and Terwijn, 1997)**.** *Some simple set is low for ML-randomness.*

An easy implication

Fact 22. *Lowness for K implies lowness for ML-randomness.*

Proof. Schnorr's Theorem relative to A states:
Z is Martin-Löf random in $A \iff$ for some b, $\forall n\ K^A(Z\restriction_n) \geq n - b$.

Thus,

- MLR can be characterized in terms of K, and
- MLR^A can be characterized in terms of K^A.

So, if A is low for K, i.e. $K =^+ K^A$, then $\mathsf{MLR} = \mathsf{MLR}^A$. □

Oracles computing an incomputable set

- For each incomputable A, the class of oracles computing A is small in the absolute sense, namely, a null class (de Leeuw et al. 1956; Sacks 1963).
- However, this class always contains a ML-random set (Kučera-Gács Theorem).
- When is this class large enough to contain a set that is even ML-random relative to A itself? (This would be the Lowness Paradigm "computed-by-many".)

Bases for ML-randomness
We will call such a set A a **base for ML-randomness**:

$$A \leq_T Z \text{ for some } Z \in \mathsf{MLR}^A.$$

They were studied by Kučera (1993).

- If A is low for ML-randomness then A is a base for ML-randomness.
- For, there is a ML-random Z such that $A \leq_T Z$.
- Then Z is ML-random relative to A.

Two theorems

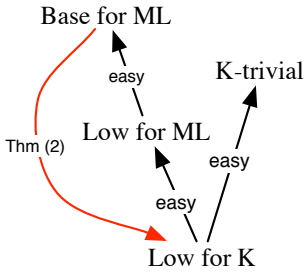
(1) There is a simple base for ML-randomness (Kučera, 1993)

(2) Each base for ML-randomness is low for K. (Hirschfeldt, N, Stephan, 2007)

- We have already seen the easy implications

$$\text{low for } K \Rightarrow \text{low for ML} \Rightarrow \text{base for ML}.$$

- Then, by Theorem (2), all three classes are the same.
- Later on we will see that this common class is a rather small subclass of the superlow sets.



(1) There is a simple base for ML-randomness
We combine two facts. (The first was mentioned earlier.)

Kučera's 1985 Theorem
Let Y be Δ^0_2 and ML-random. Then there is a simple set $A \leq_T Y$.

Lemma 23 (H, N, S 2007)**.** *Suppose $Y <_T \emptyset'$ is ML-random and $A \leq_T Y$ is c.e. Then Y is already ML-random relative to A.*

- It is now sufficient to pick as Y in Kučera's 1985 Theorem a low set that is ML-random.
- Then A is a base for ML-randomness via Y.

(2) Each base for ML-randomness is low for K

Theorem 24 (Hirschfeldt, N, Stephan 2007)**.** *Each base for Martin-Löf-randomness is low for K.*

- Recall that less injury in constructions produces sets that are closer to computable.
- The theorem shows now that the set Kučera built for his injury **free** solution to Post's problem is indeed very close to computable.

7 Far from random: K-triviality

7.1 Basics of K-trivials

Intuition on K-triviality
Recall: a set A is **K-trivial** if, for some $b \in \mathbb{N}$

$$\forall n \; K(A\restriction_n) \leq K(n) + b,$$

namely, the K complexity of all the initial segments is minimal.
This notion is opposite to ML-randomness:

- Z is ML-random iff all $K(Z\restriction_n)$ are near the upper bound $n + K(n)$;
- Z is K-trivial if the $K(Z\restriction_n)$ have the minimal possible value $K(n)$ (all within constants).

Counting theorems of Chaitin, 1976

Counting Theorem for STRINGS
For each b, at most $O(2^b)$ strings of length n satisfy $K(x) \leq K(n) + b$.

Counting Theorem for SETS
(i) For each b, at most $O(2^b)$ sets are K-trivial with constant b.
(ii) Each K-trivial set is Δ^0_2.

For the proof of the second theorem, note that
A is K-trivial via the constant $b \iff$
A is a path on the Δ^0_2 tree of strings z such that

$$K(x) \leq K(|x|) + b \text{ for each } x \preceq z.$$

Closure under effective disjoint union
$A \oplus B$ is the set which is A on the even bit positions and B on the odd positions.

The counting theorem for strings is used to prove:

Theorem 25 (Downey, H,N , Stephan 2003)**.** *If A and B are K-trivial via b, then $A \oplus B$ is K-trivial via $3b+O(1)$.*

- Idea: it is sufficient to describe each $A \oplus B \upharpoonright_{2n}$ with $K(n) + 3b + O(1)$ bits.
- We have to describe n only once; if we have a shortest description we also know its length $r = K(n)$;
- The set of strings x of length n such that $K(x) \leq r + b$ has size $O(2^b)$;
- so, we only need $b + O(1)$ bits each to describe the positions of $A \upharpoonright_n$ and $B \upharpoonright_n$ in its enumeration.

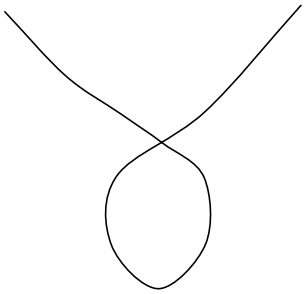
7.2 Application to Scott classes

Application to Scott classes

Definition 26. $\mathcal{S} \subseteq 2^{\mathbb{N}}$ is a **Scott class** if $\mathcal{S}$ is closed downwards under Turing reducibility, closed under joins, and each infinite binary tree $T \in \mathcal{S}$ has an infinite path in $\mathcal{S}$.

Scott classes are the standard systems of models of Peano arithmetic. They also occur in reverse mathematics as the ω-models of WKL_0.

Question of H. Friedman and McAllister
Can the Turing degrees of a Scott class look like this?



No! Scott classes are rich

Theorem 27 (Kučera, Slaman, 2007)**.** *Let $\mathcal{S}$ be a Scott class. Then, for each incomputable $X \in \mathcal{S}$, there is $Y \in \mathcal{S}$ such that Y, X are Turing incomparable.*

Proof.

- Since the complement of $\mathcal{R}_1^X$ is a Π_1^0 class relative X, we can choose $Y \in \mathcal{S}$ that is ML-random in X.
- Clearly $Y \not\leq_T X$.
- If also $X \not\leq_T Y$ we are done.

Scott classes are rich

Theorem of Kučera, Slaman, 2007
Let $\mathcal{S}$ be a Scott class. Then, for each incomputable $X \in \mathcal{S}$, there is $Y \in \mathcal{S}$ such that Y, X are Turing incomparable.

Proof continued

- Now assume $X \leq_T Y$. Then X is a base for ML-randomness, hence low for K, and hence K-trivial.
- In that case, build an infinite computable tree T such that no set $Z \in \mathit{Paths}(T)$ is K-trivial. Then $Z \not\leq_T X$ as the K-trivials are closed downward under $\leq_T$ (we show this later).
- Further, ensure that $Z \not\geq_T X$ for each $Z \in \mathit{Paths}(T)$. Since X is Δ_2^0, one can apply the Sacks preservation strategy here.

Quiz 3

(1) Why is there a low c.e. set that is not superlow?

(2) Show that each Δ_2^0 set that is low for Ω is already low for ML-randomness.

(3) Verify that each set that is low for K is K-trivial.

(4) Weak truth table reducibility $\leq_{wtt}$ is Turing reducibility with a computable bound on the maximal oracle query. Show that the K-trivial sets are closed downward under $\leq_{wtt}$.

(5) How do we define a Scott class consisting only of low sets?

8 Cost functions

Cost functions: the story

- The constructions of c.e. sets with strong lowness properties looked all very similar: building a set that is low for ML-randomness, a K-trivial, and even a simple set below a given Δ_2^0 ML-random.
- The cost function method arose to uniformize these constructions.
- Nowadays cost functions are becoming the primary tool to understand the class of K-trivials and its subclasses.
- They can be used to measure the total of changes in a computable approximation of a Δ_2^0 set (taken over all numbers).
- Lowness Paradigm 3:
  a Δ_2^0 set is close to computable if it can be approximated with a small total amount of changes.
  This is a kind-of formal version of our intuition that less injury to requirements means closer to computable.

Definition of cost functions

Definition 28. A **cost function** is a computable function

$$c : \mathbb{N} \times \mathbb{N} \to \{x \in \mathbb{Q} : \ x \geq 0\}.$$

- When building a computable approximation of a Δ^0_2 set A, we view $c(x, s)$ as the cost of changing $A(x)$ at stage s.
- We want to make the **total** cost of changes, taken over all x, **small**.

Obeying a cost function
Recall the **Limit Lemma**: A is Δ^0_2 iff $A(x) = \lim_s A_s(x)$ for some computable approximation $(A_s)_{s\in\mathbb{N}}$.

Definition 29. The computable approximation $(A_s)_{s\in\mathbb{N}}$ **obeys** a cost function c if

$$\infty > \sum_{x,s} c(x, s)\, [\![x < s \ \& \ x \text{ is least s.t. } A_{s-1}(x) \neq A_s(x)]\!].$$

We write $A \models c$ (A obeys c) if some computable approximation of A obeys c.

Usually we use this to construct some auxiliary object of finite "weight", such as a bounded request set (aka Kraft-Chaitin set), or a Solovay test.

Basic existence theorem
We say that a cost function c satisfies the **limit condition** if

$$\lim_x \sup_s c(x, s) = 0.$$

Theorem 30 (Kučera, Terwijn 1999; D,H,N,S 2003; ...)**.** *If a cost function c satisfies the limit condition, then some simple set A obeys c.*

In fact, one can achieve that $\forall B \leq_T A\, [B \models c]$, by an easy extension of the proof below.
As before we meet the $S_e : \ |W_e| = \infty \Rightarrow A \cap W_e \neq \emptyset$.

Constraint construction III
At stage s, if S_e is not satisfied yet,
look for an x, $2e \leq x < s$, such that $x \in W_e[s]$ and

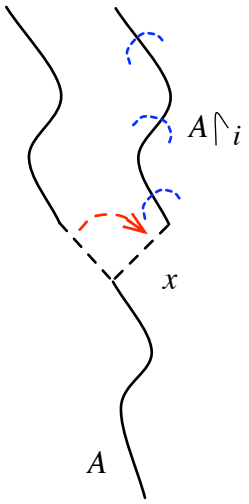
(Constraint) $\quad c(x, s) \leq 2^{-e}$.

If so, put x into A.

Does this construction have injury?
NO. One can view this construction as injury-free because c is given in advance.

In a more complicated variant, for instance the direct construction of a set that is low for K (Muchnik), the cost function c is only defined during the construction (i.e., $c(x, s)$ depends on A_{s-1}).
In such a case c can be used to emulate at stage s the restraint of individual lowness requirements (which also depends on A_{s-1}). This is like having injury.

A cost function $c_{\mathcal{K}}$ such that $A \models c \Rightarrow A$ is K-trivial

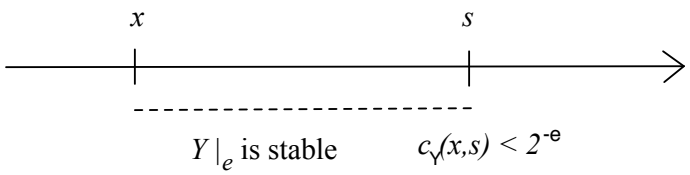


- Recall: a set A is K-trivial if $\exists b \in \mathbb{N}\, \forall n\, K(A\upharpoonright_n) \leq K(n) + b$.
- To ensure this, we build a prefix free machine M; if $A(x)$ changes then, for all i such that $s \geq i > x$, the initial segment $A\upharpoonright_i$ needs a new M-description.
- Let $\boxed{c_{\mathcal{K}}(x,s) = \sum_{i=x+1}^{s} 2^{-K_s(i)}}$
  ($K_s(x)$ is the value at stage s)
- If $A(x)$ changes at stage s then the measure of the new M descriptions is $c(x,s)$.
- So, if A obeys $c_{\mathcal{K}}$ with total cost ≤ 1 we can build the machine.
- If the total cost is $\leq 2^d$, we make the M-descriptions longer by d.

Proving Kučera's Theorem via a cost function

Kučera's 1986 Theorem
Suppose Y is a ML-random Δ^0_2 set. Then some simple set A is Turing below Y.

The proof can be phrased in the language of cost functions. Define c_Y such that, if $Y\upharpoonright_e$ is stable from x to s, then $c_Y(x,s) < 2^{-e}$.



Fact 31 (Greenberg and N; Hirschfeldt and Miller)**.** *If the Δ^0_2 set A obeys c_Y, then $A \leq_T Y$ with use function bounded by the identity.*

We build the Solovay test as follows. When $A_{s-1}(x) \neq A_s(x)$ and $c_Y(x,s) = 2^{-e}$, we list the string $Y_s\upharpoonright_e$ in $\mathcal{G}$. Since A obeys c_Y, $\mathcal{G}$ is indeed a Solovay test. Now as before one shows $A \leq_T Y$ with use bounded by the identity. □

Since Y is Δ^0_2, the cost function c_Y satisfies the limit condition.
Hence some simple A obeys c_Y. So $A \leq_T Y$.

Summary of Lecture 2

- We have studied lowness properties of Δ^0_2 sets A via randomness concepts.
- The usual lowness $A' \leq_T \emptyset'$ turns out to be a rather large class, which has interesting subclasses.
- In building a c.e. set, the less injury, the closer the set ends up to being computable.
- Kučera's injury-free construction produces a base for ML-randomness.
- Bases for ML-randomness coincide with the sets that are low for K, and the sets that are low for ML-randomness.
- K triviality is a property saying "far from random". If a set is low for K then it is K-trivial. The K-trivials are closed under $\oplus$.
- We introduced cost functions and used them to provide a direct construction of a simple K-trivial set, and to reprove Kučera's theorem.

Part III

Set theory, traceability, decanters

9 Some connections with effective descriptive set theory

9.1 Randomness via effective descriptive set theory

A version of ML-randomness based on Π^1_1 sets

- Π^1_1 sets of numbers are a high-level analog of c.e. sets, where the steps of an effective enumeration are recursive ordinals.
- Hyperarithmetical sets are the higher analog of computable sets.

X is hyperarithmetical $\iff$
both $X, \mathbb{N} - X$ are Π^1_1 $\iff$
$X \leq_T \emptyset^{(\alpha)}$ for some computable ordinal α.

Hjorth and Nies (2005) have studied the analogs, based on Π^1_1-sets, of:
string complexity K (written $\underline{K}$), and ML-randomness.

- The analog of Schnorr's Theorem holds. (The proof takes extra effort because of stages that are limit ordinals.)
- Some Π^1_1 set of numbers is $\underline{K}$-trivial and not hyperarithmetical.

The lowness properties are different now

Theorem 32 (Hjorth and Nies, 2005)**.** *If A is low for Π^1_1-ML-randomness, then A is already hyperarithmetical.*

- First we show that $\omega_1^A = \omega_1^{CK}$.
- This is used to prove that A is in fact $\underline{K}$-trivial at some $\eta < \omega_1^{CK}$, namely for some b

$$\forall n \; \underline{K}_\eta(A\upharpoonright_n) \leq \underline{K}_\eta(n) + b.$$

(Here $\underline{K}_\alpha(x)$ is the value at stage α.)
- Then A is hyperarithmetical, since the collection of Z which are $\underline{K}$-trivial at η with constant b form a hyperarithmetical tree of width $O(2^b)$.
- This is similar to Chaitin's argument to show each K-trivial is Δ^0_2.

9.2 Randomness relative to other measures on $2^{\mathbb{N}}$, and never continuously random sets

Randomness relative to a computable measure

Definition 33. A measure μ on $2^{\mathbb{N}}$ is called **continuous** if $\mu\{X\} = 0$ for each point X.

- Let μ be a continuous measure on $2^{\mathbb{N}}$ such that its representation, namely the function $[\sigma] \to \mu([\sigma])$, is **computable**.
- A typical example would be given by tossing a "biased coin", where the outcome 1 (heads) has probability $2/3$.
- We have a notion of μ-Martin-Löf-test $(G_m)_{m\in\mathbb{N}}$:
  - the open set G_m is uniformly c.e.
  - $\mu G_m \leq 2^{-m}$ for each m.

Getting back to the uniform measure within the same truth-table degree
By the following, changing the measure will make little difference for the interaction of ML-randomness with computability.

Theorem 34 (Demuth)**.** *Let μ be a computable continuous measure.*
Let Z be ML-random with respect to μ.
Then there is $\widehat{Z} \equiv_{tt} Z$ such that $\widehat{Z}$ is ML-random in the usual sense.

To prove this, one defines a total invertible Turing functional Γ such that μ is the preimage of the uniform measure λ under Γ. Now let $\widehat{Z} = \Gamma(Z)$.

"Far from random" in a strong sense: Never Continuously Random

- We consider continuous measures μ without the computability restriction.
- In this more general case, in a μ-Martin-Löf test $(G_m)_{m\in\mathbb{N}}$, the open set G_m is uniformly c.e. in the representation of μ. We now require that $\mu G_m \leq 2^{-m}$ for each m.
- (Reimann and Slaman) Let NCR be the class of sets that are not random with respect to any continuous μ. That is, for each continuous μ the set fails some μ=Martin-Löf test.

NCR sets: existence and limitations
If $\mathcal{P}$ is a countable Π^0_1 class, then $\mu(\mathcal{P}) = 0$ for any continuous μ.
This is used to prove:

Fact 35 (Kjos and Montalban 2005)**.** *Each member of a countable Π^0_1 class is never continuously random.*

In particular, $\emptyset'$ is NCR. More generally, for each computable ordinal α, the iterated jump $\emptyset^{(\alpha)}$ is NCR.
If a set is far from random in this sense, it is hyperarithmetical by the following surprising result.

Theorem 36 (Reimann and Slaman,2006)**.** *Each never continuously random set is hyperarithmetical.*

10 Traceability

Intuition on traceability
The idea of traceability: the set A is computationally weak because

- for certain functions ψ computed with oracle A, the possible values $\psi(n)$ lie in a finite set T_n of small size.
- The sets T_n are obtained effectively from n (not using A as an oracle).

Traces for functions $\omega \to \omega$ also appear in combinatorial set theory, especially forcing results related to cardinal characteristics. They are called **slaloms** there, and were introduced by Bartoszynski.

10.1 Characterizing lowness for Schnorr randomness via traceability

Computable traces

An **order function** is a function $h : \mathbb{N} \to \mathbb{N}$ that is computable, nondecreasing, and unbounded.
Example: $h(n) = \lfloor \log_2 n \rfloor$.

Definition 37. A **computable trace with bound** h is a sequence $(T_n)_{n\in\mathbb{N}}$ of non-empty sets such that

- $|T_n| \leq h(n)$ for each n
- given input n, one can compute (a strong index for) the finite set T_n.

$(T_n)_{n\in\mathbb{N}}$ is a **trace for** the function $f : \mathbb{N} \to \mathbb{N}$ if $f(n) \in T_n$ for each n.

Computable traceability

The following says A is computationally weak because it only computes functions with few possible values.

Definition 38. We say that A is **computably traceable** if there is a **fixed** h such that each function $f \leq_T A$ has a computable trace with bound h.

- Terwijn and Zambella showed that the actual choice of order function h is irrelevant (as long as it is fixed). This uses that only total functions are traced.
- Each computably traceable set is computably dominated (see next Quiz).
- Terwijn and Zambella proved that there are continuum many computably traceable sets.

Lowness for Schnorr randomness

A is **low for Schnorr randomness** if each Schnorr random set is already Schnorr random relative to A.

Theorem 39 (Terwijn, Zambella, 2001; Kjos-Hanssen, Nies, Stephan, 2005)**.** *A is computably traceable $\iff$ A is low for Schnorr randomness.*

Terwijn, Zambella characterized lowness for Schnorr tests by computable traceability;
Kjos, N, Stephan extended the result to lowness for the randomness notion.

10.2 Schnorr randomness and computable measure machines

Computable measure machines, again

Recall also that a prefix free machine M is called **computable measure machine** if its halting probability $\Omega_M = \sum_{M(\sigma)\downarrow} 2^{-|\sigma|}$ is a computable real number.

Lowness for computable measure machines

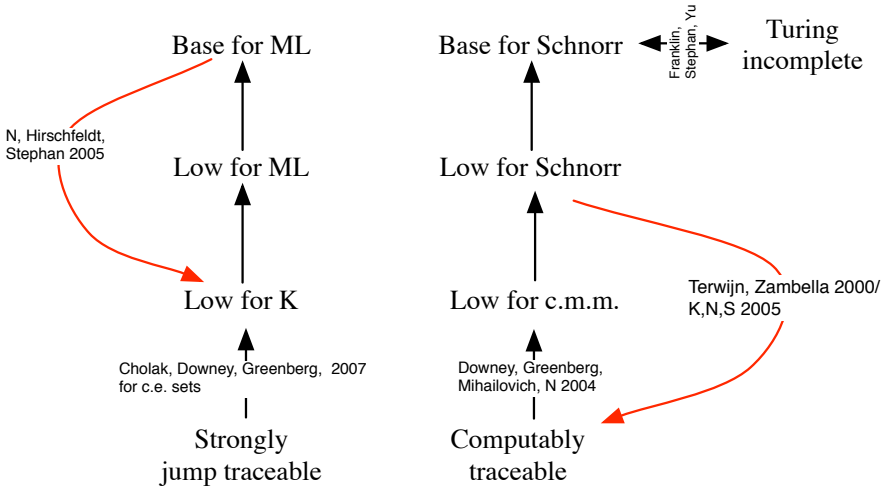
The following is a "Schnorr analog" of being low for K. The definition is more complicated here because there is no universal computable measure machine.

Definition 40. A is **low for computable measure machines** if for each computable measure machines M^A relative to A, there is a computable measure machines N such that

$$\forall x\, K_N(x) \leq^+ K_{M^A}(x).$$

Theorem 41 (Downey, Greenberg, Mihailovich, Nies 2005)**.** *A is computably traceable $\iff$ A is low for computable measure machines.*

Comparing implications: Martin-Löf vs. Schnorr



10.3 Strong jump traceability

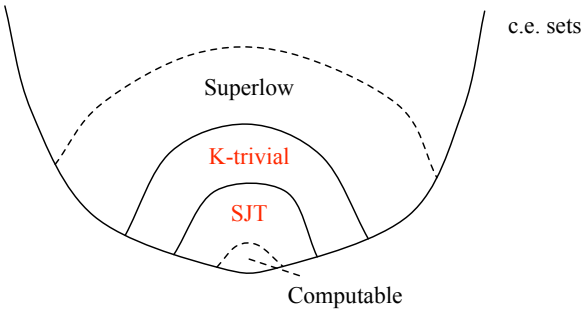
Strongly jump traceable sets

- A **computably enumerable trace with bound** h is a uniformly computably enumerable sequence $(T_x)_{x\in\mathbb{N}}$ such that $|T_x| \leq h(x)$ for each x.
- Recall that $J^A(x)$ is the value on input x of a universal A-partial computable function.
- The set A is called **strongly jump traceable** if for **each** order function h, there is a c.e. trace $(T_x)_{x\in\mathbb{N}}$ with bound h such that, whenever $J^A(x)$ is defined, we have $J^A(x) \in T_x$. This notion was introduced by Figueira, N, Stephan 2004.
- Unlike the case of computable traceability, here it matters that we require each order function h as a bound for some trace.
  Jump traceability, where one merely requires that the tracing works for **some** bound h, is a much weaker notion.

Comparing K-trivial and SJT within the c.e. sets

The c.e. strongly jump traceable sets form a **proper** subclass of the c.e. K-trivial sets by Cholak, Downey, Greenberg 2006.

- Both classes are closed downward under $\leq_T$.
- Both classes are closed under $\oplus$.
- The c.e. K-trivials have a Σ^0_3 index set;
  the c.e. SJTs have a Π^0_4-complete index set (Selwyn Ng).



Theorem 42 (Figueira, N, Stephan 2004)**.** *There is a simple strongly jump traceable set.*

- Recall the construction of a low simple set: a simplicity requirement S_e can only injure a lowness requirement L_i if $e < i$.
- This yields a trace $(T_i)_{i\in\mathbb{N}}$ with bound $i+1$: the computation $J^A(i)$ becomes undefined for at most i times; let T_i be the set of its values during the construction.
- To obtain smaller traces, we allow much less injury to the L_i.
- Let h be a function that is computably approximable from above, unbounded, and dominated by all order functions g.
- At stage s, requirement S_e can only injure L_i if $h_s(e) < i$.
- Then, for each order function g, there is a trace with bound g for J^A.

Do we need the injury at all?

- Can we build a strongly jump traceable c.e. set without **any** lowness requirements and injury to them?
- Given the SJTs are even closer to computable than the K-trivials, we would expect so.
- By definition, the SJTs are weak as oracles. To answer the question, we need to characterize the SJT's via the "few-changes", and via the "computed-by-many" lowness paradigms.

SJT's and the "few-changes" paradigm

Recall: a **cost function** is a computable function $c : \mathbb{N}\times\mathbb{N} \to \{x \in \mathbb{Q}\colon\ x \geq 0\}$.

- We say c is **monotonic** if $c(x, s)$ is nonincreasing in x and nondecreasing in s.
- A monotonic cost function c is **benign** if there is a computable bound $g(n)$ on the length of any sequence $x_0 < x_1 < \ldots$ such that $c(x_i, x_{i+1}) \geq 2^{-n}$ for each i.
- For instance, the cost function for K-triviality is benign via $g(n) = 2^n$.

Theorem 43 (Greenberg and Nies, to appear)**.** *Let A be c.e. Then*
A is strongly jump traceable $\iff$ A obeys each benign cost function.

In particular, SJT implies K-trivial for c.e. sets (C,D,G 2006).

Quiz 4

(1) Show that computably traceable $\Rightarrow$ computably dominated.

(2) Show the implication is proper.

(3) Show that the standard cost function $c_{\mathcal{K}}$ is benign.

SJT's and the "computed-by-many" paradigm

- Recall that Y is ω-c.e. if Y is Δ^0_2 with a computably bounded number of mind changes.
- It is easy to obtain a ML-random ω-c.e. set: for instance Chaitin's number Ω, or a superlow ML-random.
- The following theorem says that a c.e. set A is strongly jump traceable iff it is computed by many ML-random oracles.

Theorem 44 (Greenberg, Hirschfeldt and Nies, to appear)**.** *Let A be c.e. Then A is strongly jump traceable $\iff$ A is Turing below each ω-c.e. ML-random set.*

"$\Rightarrow$" follows from the previous result: if Y is ω-c.e. then its associated cost function c_Y is benign. Since $A \models c_Y$ and Y is ML-random, we obtain $A \leq_T Y$.
"$\Leftarrow$" is harder.

Lower bounds for null Σ^0_3 classes of random sets

Theorem 45 (Hirschfeldt/Miller)**.** *For each null Σ^0_3 class $\mathcal{H}$ of ML-random sets, there is a simple set A such that $A \leq_T Y$ for each $Y \in \mathcal{H}$.*

- If $\mathcal{H}$ consists of a single ML-random Δ^0_2 set we have Kučera's 1986 Theorem.
- The general theorem is proved by defining a cost function $c_{\mathcal{H}}$ with the limit condition.
- Whenever a c.e. set A obeys $c_{\mathcal{H}}$, then $A \leq_T Y$ for each $Y \in \mathcal{H}$.
- Recall that some simple set A obeys $c_{\mathcal{H}}$. □

Building an SJT without injury

- Now let $\mathcal{H}$ be the countable Σ^0_3 class of ω-c.e. ML-randoms.
- The simple lower bound A must be SJT. It was constructed without injury. So we have built an SJT without injury.

SJT's and randomness

- Recall that Z is superlow if $Z' \leq_{tt} \emptyset'$. Dually, we say that Z is superhigh if $\emptyset'' \leq_{tt} Z'$.
- A random set can be superlow (low basis theorem). It can also be superhigh but Turing incomplete (Kučera coding).

Via randomness we can characterize the c.e. strongly jump traceable sets in various ways.

Theorem 46 (Greenberg, Hirschfeldt and Nies, to appear)**.** *A c.e. set A is strongly jump traceable*
$\iff$ *A is Turing below each super**low** ML-random set*
$\iff$ *A is Turing below each super**high** ML-random set .*

Comparing K-trivial and SJT sets, again

- Both classes have characterizations via all three lowness paradigms (for SJT sets, so far this only works in the c.e. case).
- The K-trivials form the smallest known subclass of the Δ^0_2 sets we can get to via a "low-for-something" definition.
- It is a persistent **open question** whether each K-trivial is below a Turing incomplete ML-random set. (Informally, we don't know whether Kučera's injury free solution to Post's problem is equivalent to building a K-trivial.)
- In contrast, each c.e. SJT is below **each** superlow ML-random.
- This suggests that SJT is a very small subclass of the K-trivials.

Lowness paradigms for K-trivial and SJT

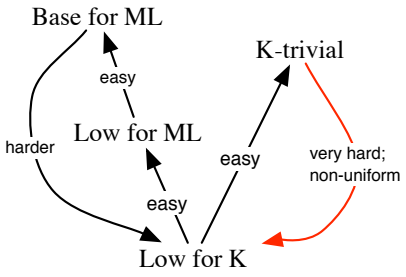
Let $\mathcal{H}^{\Diamond}$ = the c.e. sets A Turing below each ML-random set in $\mathcal{H}$.

	Few changes:	Weak as an oracle:	Computed by many oracles:
K-trivial	obeying the standard cost function $c_{\mathcal{K}}$	low for K; low for ML-rd; low for weak 2-rd;	Base for ML-randomness
strongly jump-traceable	**for c.e. sets:** obeying all benign cost functions	the very definition	**for c.e. sets:** (ω-c.e.)$^{\Diamond}$; superlow$^{\Diamond}$; superhigh$^{\Diamond}$;

11 Each K-trivial is low for K

Remaining implication: K-triviality $\Rightarrow$ lowness for K

Theorem 47. *Each K-trivial set is low for K.*



This extends the result of D,H,N,S (2003) that K-trivials are Turing incomplete. The theorem was obtained by Hirschfeldt and Nies (2005), via a modification of Nies' previous result that the K-trivial sets are closed downward under $\leq_T$.

The Machine Existence Theorem

We use this tool:

- A c.e. set $L \subseteq \mathbb{N} \times \{0,1\}^*$ is a **bounded request set** if

$$1 \geq \sum_{r,y} 2^{-r} [\![\langle r, y\rangle \in L]\!].$$

- From a bounded request set L, one can obtain a prefix free machine M such that

$$\langle r, y\rangle \in L \Rightarrow M(\sigma) = y \text{ for some } \sigma \text{ such that } |\sigma| = r.$$

wtt-incompleteness

The downward closure of the class $\mathcal{K}$ of K-trivials under $\leq_{wtt}$ was an exercise in Quiz 3. Since the wtt-complete set Ω is not K–trivial, no K–trivial set A satisfies $\emptyset' \leq_{wtt} A$. We first give a **direct** proof of this. Suppose $\emptyset' \leq_{wtt} A$ for K-trivial A.

- We build a c.e. set B, and by the Recursion Theorem we can assume we are given a total wtt-reduction Γ such that $B = \Gamma^A$, whose use is bounded by a computable function g.
- We build a bounded request set L. Thus we enumerate requests $\langle r, n\rangle$ and have to ensure $\sum_r 2^{-r}$ is at most 1. By the recursion theorem, we may assume the coding constant d for L is given in advance. Then, putting $\langle r, n\rangle$ into L causes $K(n) \leq r + d$ and hence $K(A\upharpoonright_n) \leq r + b + d$, where b is the triviality constant.

The number of levels

- The construction has k levels, where

$$k = 2^{b+d+1}$$

- Let $n = g(k)$ (the use bound). We wait till $\Gamma^A(k)$ converges, and put the single request $\langle r, n\rangle$ into L, where $r = 1$. Our total investment is $1/2$.
- Each time the "opponent" has a description σ of length $\leq r + b + d$ such that $\mathbb{U}(\sigma) = A\upharpoonright_n$, we force $A\upharpoonright_n$ to change, by putting into B the largest number $\leq k$ which is not yet in B.
- If we carry out $k + 1$ such A changes, then the opponent's total measure of descriptions σ is

$$(k + 1)2^{-(1+b+d)} > 1,$$

contradiction.

Turing-incompleteness

- Consider the more general result that K-trivial sets are T-incomplete (Downey, Hirschfeldt, N, Stephan, 2003). There is no computable bound on the use of $\Gamma^A(k)$.
- The problem now is that the opponent might, before giving a description of $A_s\upharpoonright_n$, move this use beyond n. This deprives us of the possibility to cause further changes of to$A\upharpoonright_n$.
- The solution is to carry out **many** attempts in parallel, based on computations $\Gamma^A(m)$ for different m.
- Each time the use of such a computation changes, the attempt is cancelled. What we placed in L for this attempt now becomes **garbage**. We have to ensure that the weight of the garbage does not build up too much, otherwise L is not a bounded request set.

More details: j-sets

The following is a way to keep track of the number of times the opponent had to give new descriptions of strings $A_s\restriction_n$.

Technical detail: we only consider *stages* (italics) at which A looks K-trivial with constant b up to the previous *stage*.

- At *stage* t, a finite set E is a j-set if for each $n \in E$
  - first we put a request $\langle r_n, n\rangle$ into L, and then
  - for j times at *stages* $s < t$ the opponent had to give new descriptions of $A_s\restriction_n$ of length $r_n + b + d$.
- A c.e. set with an enumeration $E = \bigcup E_t$ is a j-set if E_t is a j-set at each *stage* t.

The weight of a k-set

The **weight** of a set $E \subseteq \mathbb{N}$ is $\sum 2^{-r_n}\ [\![n \in E]\!]$.

Fact 48. *If the c.e. set E is a k-set, $k = 2^{b+d+1}$ as defined above, then the weight of E is at most $1/2$.*

The reason for this: k times the opponent has to match
our description of n, which has length r_n, by
a description of a string $A_s\restriction_n$ that is by at most $b + d$ longer.

Procedures

Assume A is K-trivial and Turing complete. As in the wtt-case, we attempt to build a k-set F_k of weight $> 1/2$ to reach a contradiction.

- Procedure P_j ($2 \leq j \leq k$) enumerates a j-set F_j.
- The construction begins by calling P_k, which calls P_{k-1} many times, and so on down to P_2, which enumerates L (and F_2).

Each procedure P_j has rational parameters $q, \beta \in [0, 1]$.

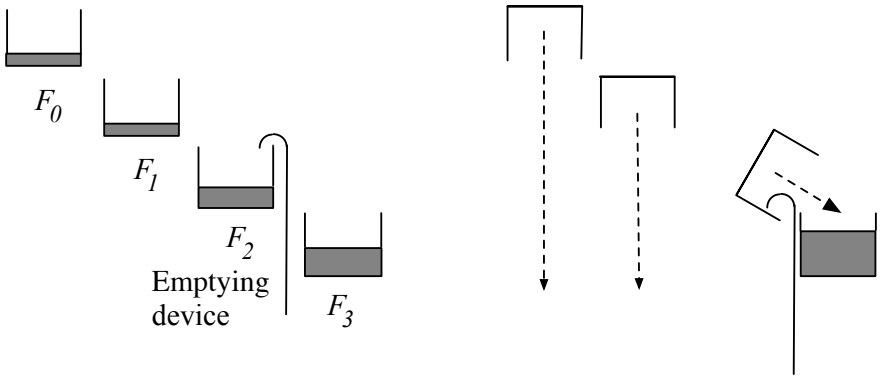
- The **goal** q is the weight it wants F_j to reach.
- The **garbage quota** β is how much garbage it can produce (details on garbage later).

When a procedure reaches its goal it returns.

Decanter model

- We visualize this construction by a machine consisting of decanters
$$F_k, F_{k-1}, \ldots, F_0.$$
At any *stage* F_j is a j set. F_{j-1} can be emptied into F_j.
- The procedure $P_j(q, \beta)$ wants F_j to reach a weight of q.
It fills F_{j-1} up to q and then returns, by emptying it into F_j.
The emptying is done by enumerating m_j into B and hence adding one more A-change.
- The emptying device is a hook (the $\gamma^A(m)$-marker). It is used on purpose when we put m_j into B.
- It may go off finitely often by itself when A-changes "prematurely".
- When F_{j-1} is emptied into F_j then $F_{j-2}, \ldots, F_0$ are spilled on the floor.

The decanter machine in action



- The recursion starts by calling P_k with goal 1.
- It calls P_{k-1} etc down to P_0.
- Thus, wine is first poured into the highest decanter F_0 (and goes into the left domain of L).
- We want to ensure that at least half the wine we put into F_0 reaches F_k.

Here is a key feature to achieve this:

- When we have to cancel a run $P_j(q, \beta)$ because of a premature A-change, what becomes garbage is **not** F_{j-1}, but rather what the sub-procedures called by this run were working on.
- For the set F_{j-1} already is a $j-1$-set, so all we need is another A-change, which is provided here by the cancellation itself as opposed to being caused actively once the run reaches its goal.

Recall that the parameter β is the amount of garbage $P_j(q, \beta)$ allows. If v is the number of times an emptying device has gone off by itself, then P_j lets P_{j-1} fill F_{j-1} in portions of $2^{-v}\beta$. Then when F_{j-1} is emptied into F_j, at most $2^{-v}\beta$ can be lost because of being in higher decanters $F_{j-2}, \dots, F_0$.

$P_2(q, \beta)$

- The bottom procedure $P_2(q, \beta)$ enumerates L. This is where the recursion reaches ground.
- $P_2(q, \beta)$ puts requests $\langle r_n, n\rangle$ into L and the top decanter F_0, where $2^{-r_n} = 2^{-v}\beta$.
- Once it sees the corresponding $A\restriction_n$ description, it empties F_0 into F_1.
- However, if the hook $\gamma^A(m)$ belonging to P_2 moves before that, then F_0 is spilled on the floor, while F_1 is emptied into F_2.

So much for the discussion of Turing incompleteness.

Each K-trivial set is low

We now want to computably approximate whether $J^A(e)\downarrow$.

- For $j < k$, a procedure $P_j(q, \beta)$ is called by P_{j+1} when $J^A(e)$ newly converges. Its goal q is $\alpha 2^{-e}$, where α is the garbage quota of the procedure of type P_{j+1}.
- Procedures P_j run in parallel for different e. So we now have a tree of decanters.
- There is no emptying device: we cannot change A actively any more. To achieve lowness, we are happy if it doesn't.
- However, this creates a new type of garbage, where $P_j(q, \beta)$ reaches its goal, but no A change happens after, which would be needed to empty an F_{j-1} into F_j. The total garbage weighs at most $\sum_e 2^{-e}\alpha$, which is ok.

The golden run

- The initial procedure P_k never returns, since it has goal 1, while a k-set has weight at most $1/2$.
- So there must be a **golden run** of a procedure $P_{j+1}(q, \alpha)$:

  it doesn't reach its goal, but all the subprocedures it calls either reach their goals, or are cancelled by a premature A change.
- The golden run shows that A is low:

  When the run of the subprocedure P_j based on a computation $J^A(e)$ returns, then we guess that $J^A(e)$ **converges**.

  If A changes then P_{j+1} receives the fixed quantity $2^{-e}\alpha$. In this case we change the guess back to **"divergent"**.

  This can only happen $2^e q/\alpha$ times, else P_{j+1} reaches its goal.

Superlowness, and non-uniformity

- The argument actually shows that A is **superlow**: the number of mind changes in the approximation of A' is computably bounded.
- The lowness index (i.e. Turing reduction for $A' \leq_T \emptyset'$) is **not** obtained effectively in the index for A and the constant for K-triviality. To obtain this index, we needed to know which run is golden.
- This non-uniformity is necessary by **Downey, Hirschfeldt, N, Stephan 2003**.
- K-trivial $\Rightarrow$ low for K is non-uniform for the same reason. We can't compute the constant for lowness for K from and index for A and the K-triviality constant.
- This may be why attempts at giving "combinatorial", half-page proofs have failed so far.

Proving the full result

We now show that **K-trivial $\Rightarrow$ low for K**.

- For $j < k$ a procedure $P_j(q, \beta)$ is started when $\mathbb{U}^A(\sigma) = y$ newly converges. Its goal q is $\alpha 2^{-|\sigma|}$.
- We call, in parallel, procedures based on different inputs σ. So we again have a tree of decanters.
- Via a golden run P_{j+1}, we show that A is low for K. When $P_j(q, \beta)$ associated with $\mathbb{U}^A(\sigma) = y$ returns, we have the right to enumerate a request $\langle |\sigma| + c, y\rangle$ into a set W (c is some constant).
- Every A change means useless weight in W, since we issued request for a wrong computation $\mathbb{U}^A(\sigma)$. The fact that P_{j+1} does not reach its goal implies that the total weight is bounded. Hence W is a bounded request set.

Cost function characterization of the K-trivials
Recall: the cost function $c_{\mathcal{K}}$ for K-triviality is given by

$$c_{\mathcal{K}}(x,s) = \sum_{i=x+1}^{s} 2^{-K_s(i)}.$$

We could also use $c(x,s) = \text{Prob}[\{\sigma\colon\ \mathbb{U}_s(\sigma) \geq x\}]$, the chance that the universal machine prints a string $\geq x$ within s steps.

Theorem 49 (Nies 05)**.** *A is K-trivial $\iff$*
some computable approximation of A obeys $c_{\mathcal{K}}$.

We have already done "$\Leftarrow$". For "$\Rightarrow$" one uses the golden run.

Corollary 50. *For each K-trivial A there is a* **c.e.** *K-trivial set $D \geq_{tt} A$.*

D is the **change set** $\{\langle x,i\rangle\colon\ A(x)$ changes at least i times$\}$. One verifies that D obeys $c_{\mathcal{K}}$ as well.
Actually, this works for any cost function in place of $c_{\mathcal{K}}$.

Summary of Lecture 3

- We have studied versions of ML-randomness and K based on Π^1_1 sets.
- We have defined the NCR sets, a further notion of being far from random.
- Traceability is a meta-concept leading to interesting lowness properties.
- Computable traceability characterizes lowness for Schnorr randomness.
- The c.e. strongly jump traceable sets form an interesting, very small subclass of the c.e. K-trivial sets. They can be naturally characterized via the "computed-by-many" paradigm.
- The decanter method shows each K-trivial is Turing incomplete. Adding the non-uniform golden run method shows that each K-trivial is in fact low for K.