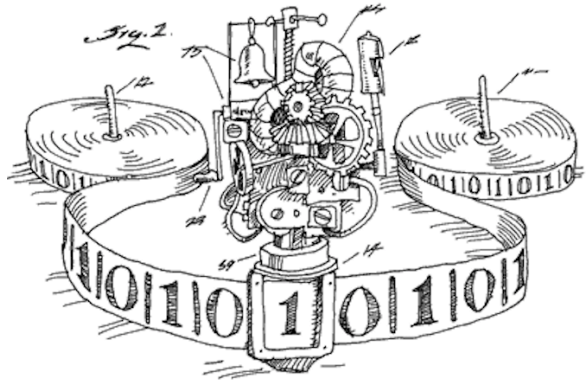
Coding information into random reals



by George Barmpalias

joint work with Lewis-Pye and Teutsch

December 6, Oaxaca

Victoria University of Wellington

Chinese Academy of Sciences

Optimal redundancy in computations from random oracles *

George Barmpalias Andrew Lewis-Pye

Lower bounds on the redundancy in computations from random oracles via betting strategies with restricted wagers *

George Barmpalias Andrew Lewis-Pye Jason Teutsch

Information vs randomness

- Information is structure, regularity
- Randomness is lack of structure, noise.

However some times we can obfuscate information.

We make it look like noise ...

...while still being able to extract it

from what is apparently a random object.

We study the limits of this phenomenon in the context of...

Algorithmic Randomness

Randomness with respect to effective processes.

Frameworks:

- Measure (effective statistical tests)
- Incompressibility (Kolmogorov complexity)
- Effective betting strategies (martingales)

Strengths of randomness correspond to ...

...complexity requirements of processes.

Too much randomness: information extraction impossible

The plan of the talk

(1) Finite/Infinite examples of information vs randomness.

(2) Our problem: optimizing the redundancy.

(3) Existing works on this topic.

(4) Establishing optimal lower bounds.

(5) The main result: optimal coding.

(6) The known method(s).

(7) The new coding method.

(8) Verification of the new coding method.

Finite example

Prefix-free machines as decompressors

Prefix-free machines as information extractors

$\mathrm{M}(\sigma) = \tau$: σ describes τ

Shortest descriptions are random: $\mathrm{K}(\sigma) \geq |\sigma|$

Most strings are very random: $\mathrm{K}(\sigma) \geq |\sigma| + \mathrm{K}(|\sigma|)$

Very random strings cannot be decompressed into anything.

If σ is very random then $\mathrm{M}(\sigma)$ does not converge.

Infinite examples

Π^0_2-randoms do not compute the halting problem

Π^0_2-randoms do not compute any complete extension of PA

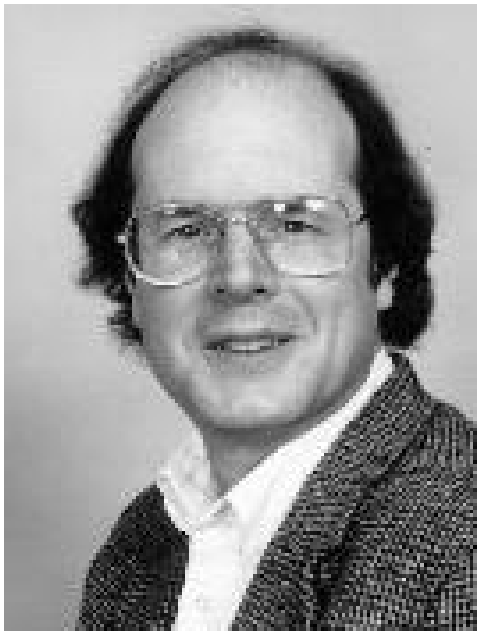
Π^0_2-randoms do not compute anything incomputable that is computed by the halting problem.

Martin-Löf randomness is uniformly Π^0_2-randomness

It has the most interesting interactions with computablility

Kucera-Gács theorem: Every sequence is computable from a Martin-Löf random sequence.

Answers Charles Bennett's question



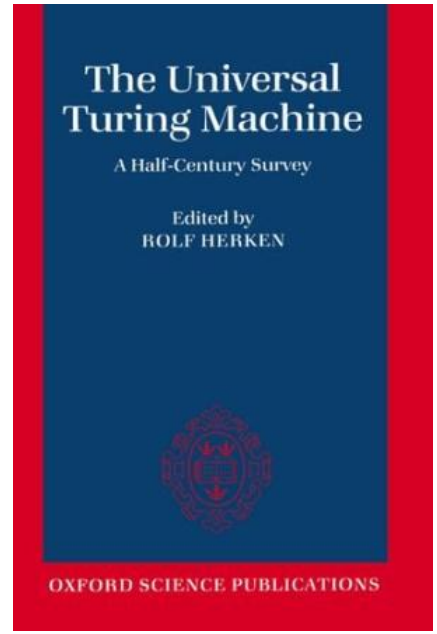
Logical Depth and Physical Complexity

Charles H. Bennett

Abstract. Some mathematical and natural objects (a random sequence, a sequence of zeros, a perfect crystal, a gas) are intuitively trivial, while others (e.g., the human body, the digits of π) contain internal evidence of a nontrivial causal history.

We formalize this distinction by defining an object's "logical depth" as the time required by a standard universal Turing machine to generate it from an input that is algorithmically random (i.e., Martin-Löf random). This definition of depth is shown to be reasonably machine-independent, as well as obeying a slow-growth law: deep objects cannot be quickly produced from shallow ones by any deterministic process, nor with much probability by a probabilistic process, but can be produced slowly.

Next we apply depth to the physical problem of "self-organization", inquiring in particular under what conditions (e.g., noise, irreversibility, spatial and other symmetries of the initial conditions and equations of motion) statistical-mechanical model systems can imitate computers well enough to undergo unbounded increase of depth in the limit of infinite space and time.



This is the infinite analog of far more obvious fact that every finite string is computable from an algorithmically random string (e.g. its minimal program).

Finite/Infinite analogies – string vs stream coding

Notion	Finite	Infinite
Source	σ	X
Code	σ^*	Y
Code-length	$\lvert\sigma^*\rvert$	$n \mapsto f(n)$
Optimal code	$K(\sigma)$	?

Finite/Infinite analogies – string vs stream coding

Notion	Finite	Infinite
Source	σ	X
Code	σ^*	Y
Code-length	$\lvert\sigma^*\rvert$	$n \mapsto f(n)$
Optimal code	$K(\sigma)$	$n \mapsto K(X \upharpoonright_n)?$

Finite/Infinite analogies – string vs stream coding

Notion	Finite	Infinite
Source	σ	X
Code	σ^*	Y
Code-length	$\lvert\sigma^*\rvert$	$n \mapsto f(n)$
Optimal code	$K(\sigma)$	~~$n \mapsto K(X_{\lceil n})$~~

Finite/Infinite analogies – string vs stream coding

Notion	Finite	Infinite
Source	σ	X
Code	σ^*	Y
Code-length	$\lvert\sigma^*\rvert$	$\mathrm{n} \mapsto \mathrm{f}(\mathrm{n})$
Optimal code	$\mathrm{K}(\sigma)$	$\mathrm{f} \geq \mathrm{K}(\mathrm{X}\restriction_{\mathrm{n}})$

Computing from random sequences

How many extra bits of a random do we need in order to compute the first n bits of a sequence in general?

Kucera achieved redundancy $n \log n$ in this reduction.

Gács achieved redundancy $3\sqrt{n} \cdot \log n$ via block-coding.

Redundancy cannot be $O(1)$ (Downey/Hirschfeldt).

What is the optimal redundancy ?

Reprinted from INFORMATION AND CONTROL
All Rights Reserved by Academic Press, New York and London

Vol. 70, No. 2/3, August/September 1986
Printed in Belgium

Every Sequence Is Reducible to a Random One

PÉTER GÁCS*

Computer Science Department, Boston University,
Boston, Massachusetts 02215

Every infinite sequence is Turing–reducible to an infinite sequence which is random in the sense of Martin–Löf.

INTRODUCTION

Charles Bennett asked whether every infinite binary sequence can be obtained from an "incompressible" one by a Turing machine. He proved that this is the case for arithmetical sequences. The question has some philosophical interest because it permits us to view even very pathological sequences as the result of the combination of two relatively well-understood processes: the completely chaotic outcome of coin-tossing, and a transducer algorithm.

THEOREM. *Let E be a constructive closed set with $\lambda(E) > 0$. Then there exists a process F such that $F(E) = B$. Moreover, there is a constant c such that on every nonterminal string x of length n we have*

$$|F(x)| \geqslant n - 3\sqrt{n}\log n + c.$$

The last property of F says that we need no more than $3\sqrt{n}\log n$ bits of redundant information in our uniform generation of arbitrary sequences from random ones.

THE JOURNAL OF SYMBOLIC LOGIC
Volume 69, Number 3, Sept. 2004

ON THE CONSTRUCTION OF EFFECTIVELY RANDOM SETS

WOLFGANG MERKLE AND NENAD MIHAILOVIĆ

Proof by martingales...

...yes we have redundancy $o(n)$.

Every Sequence is Decompressible from a Random One

David Doty *

Department of Computer Science, Iowa State University, Ames, IA 50011, USA.
ddoty *at* iastate *dot* edu

Abstract. Kučera and Gács independently showed that every infinite sequence is Turing reducible to a Martin-Löf random sequence. We extend this result to show that every infinite sequence S is Turing reducible to a Martin-Löf random sequence R such that the asymptotic number of bits of R needed to compute n bits of S, divided by n, is precisely the constructive dimension of S. We show that this is the optimal ratio of query bits to computed bits achievable with Turing reductions. As an application of this result, we give a new characterization of constructive dimension in terms of Turing reduction compression ratios.

B. Ya. Ryabko. Noiseless coding of combinatorial sources. *Problems of Information Transmission*, 22:170–179, 1986.

Computing from random sequences

How many extra bits of a random do we need in order to compute the first n bits of a sequence in general?

Kucera achieved redundancy $n \log n$ in this reduction.

Gács achieved redundancy $3\sqrt{n} \cdot \log n$ via block-coding.

Redundancy cannot be $O(1)$ (Downey/Hirschfeldt).

What is the optimal redundancy ?

Optimal lower bounds

Barmpalias/Lewis/Teutsch, Information and Computation 2016:

> Suppose that g is a nondecreasing computable function such that $\sum_i 2^{-g(i)} = \infty$.
>
> Then there exists a sequence which is not computable from any random sequence with use $n + g(n)$.

Proof by effective martingales with controlled wager granularity.

Turing reductions $\rightarrow$ betting strategies.

Smaller oracle-use $\rightarrow$ larger wagers.

Lower bound by betting strategies with restricted wagers

Betting strategies are often formalized by martingales.

A strategy with restricted wagers is only allowed to bet an amount from a specified set of values at each step.

Integer-valued martingales is a well-studied example.

A g-granular martingale bets multiples of $2^{-g(s)}$ at step s.

If g is computable and $\sum_i 2^{-g(i)} = \infty$ given any g-granular supermartingale M, there is a real X and a supermartingale N such that N succeeds on X but M doesn't succeed on X.

Reductions Φ with tight use and martingales

Turing functional Φ induces a supermartingale.

Low use in the Turing functional corresponds to decreased granularity on the martingale.

Decreased granularity allows non-randoms to succeed.

If a real succeeds in the Φ-martingale, then the complexity of the image affects the complexity of the oracle that maps to it.

A dip in the complexity of the image creates a dip in the complexity of the oracle.

If the Φ-martingale succeeds on a non-random X then any Y that Φ-maps to X is non-random.

Optimal lower bounds

Barmpalias/Lewis/Teutsch, Information and Computation 2016:

> Suppose that g is a nondecreasing computable function such that $\sum_i 2^{-g(i)} = \infty$.
>
> Then there exists a sequence which is not computable from any random sequence with use $n + g(n)$.

Proof by effective martingales with controlled wager granularity.

Turing reductions $\rightarrow$ betting strategies.

Smaller oracle-use $\rightarrow$ larger wagers.

Conjecture

Suppose that g is a nondecreasing computable function such that $\sum_i 2^{-g(i)} < \infty$.

Then every sequence is computable from a random one with redundancy $g(n) + O(1)$.

A new coding method into random sequences is needed....

...for any improvement of the known bounds.

Theorem (Barmpalias and Lewis 2016)

Suppose that g is a nondecreasing computable function such that $\sum_i 2^{-g(i)} < 1$.

Then every sequence is computable from a random one with redundancy $g(n)$.

A new coding method into random sequences was needed....

...for any improvement of the known bounds.

If $\mathcal{P}$ is a Π^0_1 class and $\sum_i 2^{-g(i)} < \mu(\mathcal{P})$ then every real is computable from a member of $\mathcal{P}$ with redundancy g.

The known method (simple form)

(1) Start with a Π^0_1 class $\mathscr{P} \neq \emptyset$ which only contains randoms.

(2) Choose the length ℓ_i of the block which will code bit i.

(3) The oracle use for the first n bits will be $L_n = \sum_{i<n} \ell_i$.

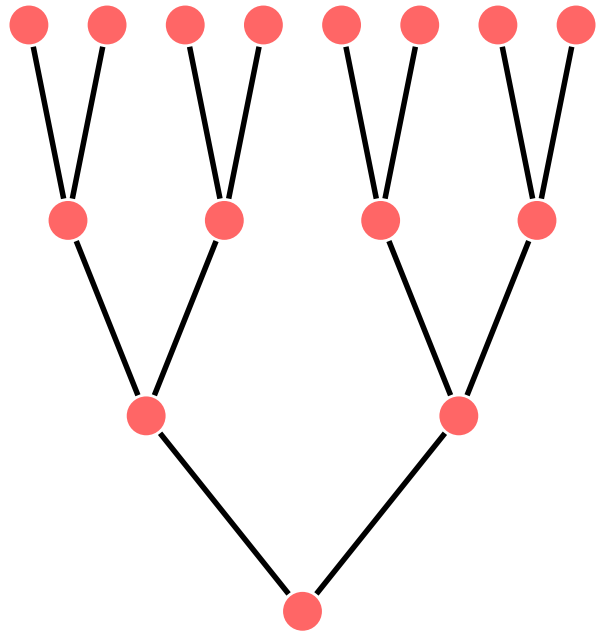
(4) Form the subclass $\mathscr{P}^*$ of $\mathscr{P}$ with the property that

> for each $X \in \mathscr{P}^*$ and each n, there are at least 2 extensions of $X \upharpoonright_{L_n}$ of length L_{n+1} in $X \in \mathscr{P}^*$.

and ...

(5) Hope that $\mathscr{P}^* \neq \emptyset$ (due to the growth of (ℓ_i)).

How does the code-tree look like?



Isomorphic to the full binary tree.

What is the required growth of (ℓ_i)?

If $\sum_i 2^{-\ell_i} < \infty$ then you can find $\mathcal{P}$ with $\mathcal{P}^* \neq \emptyset$

If $\sum_i 2^{-\ell_i} = \infty$ then $\mathcal{P}^* = \emptyset$

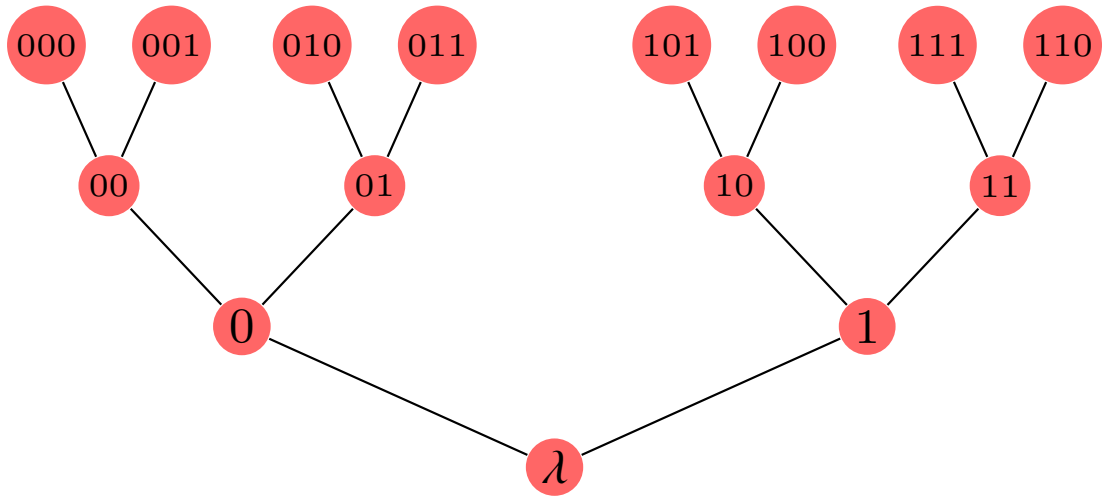
What oracle-use does this condition allow?

$$L_n = \sum_{i<n} \log i = \log n! \sim n \cdot \log n.$$

Why so bad?

Logarithmic overheads are pilled on the top of one another.

We pay a price for a good-looking code-tree



→ Every string has a unique code.

→ If Y codes X then $Y \upharpoonright_{L_n}$ is computable from $X \upharpoonright_n$ and $\emptyset'$.

Why don't we code in blocks? (Gács)

(1) Overheads concern the coding steps and not the coded bits.

(2) If I code 200 bits in two steps, I only need $2 \log 2$ overhead.

(3) At step n code m_n more bits with $\log n$ added overhead.

(4) Lets make m_n grow like crazy!

...not so fast...

In order to compute bit $m_0 + m_1 + 1$ we need to compute bits $m_0 + m_1 + j$, $j < m_2$.

and ...

(5) This is an extra overhead from the source block-lengths.

The known method (with block-coding)

(1) Start with a Π^0_1 class $\mathscr{P} \neq \emptyset$ which only contains randoms.

(2) Choose the length m_i of the block coded at step i.

(3) Choose the length $\ell_i = m_i + g(i)$ for the ith block.

(4) The oracle-use for the first $M_n = \sum_{i<n} m_i$ bits is $L_n = \sum_{i<n} \ell_i$.

(5) Form the subclass $\mathscr{P}^*$ of $\mathscr{P}$ with the property that

> for each $X \in \mathscr{P}^*$ and n, there are at least 2^{m_n} extensions of $X \upharpoonright_{L_n}$ of length L_{n+1} in $X \in \mathscr{P}^*$.

and ...

(5) Hope that $\mathscr{P}^* \neq \emptyset$ (due to the growth of (ℓ_i)).

What is the required growth of (ℓ_i)?

If $\sum_i 2^{m_i - \ell_i} < \infty$ then you can find $\mathcal{P}$ with $\mathcal{P}^* \neq \emptyset$

If $\sum_i 2^{m_i - \ell_i} = \infty$ then $\mathcal{P}^* = \emptyset$

What oracle-use does this condition allow for the bits in

$$[M_n, M_n + m_n)?$$

$$L_n = M_n + m_n + \sum_{i<n} \log i \sim M_{n+1} + n \cdot \log n.$$

What choice of (m_i) minimizes the oracle-use?

What is the required growth of (ℓ_i)?

If $\sum_i 2^{-g(i)} < 1$ and

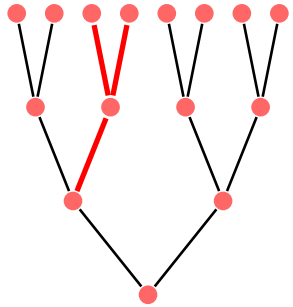
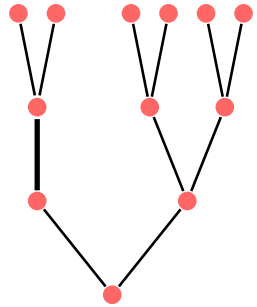
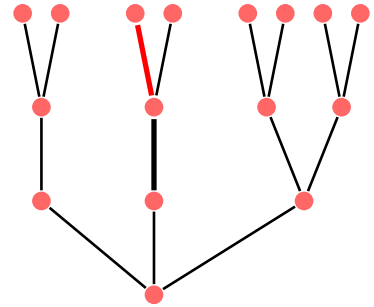
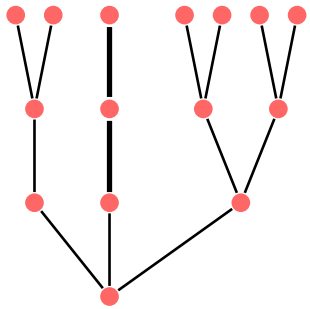
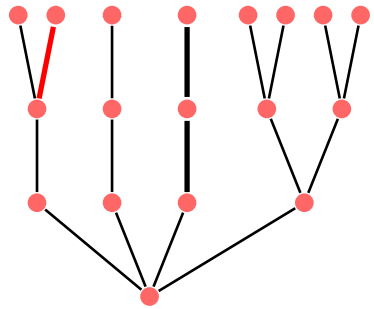
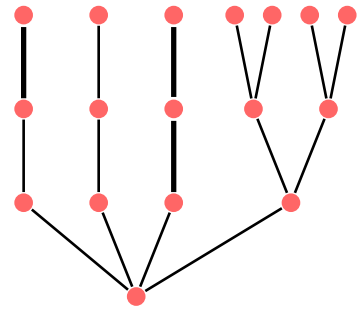
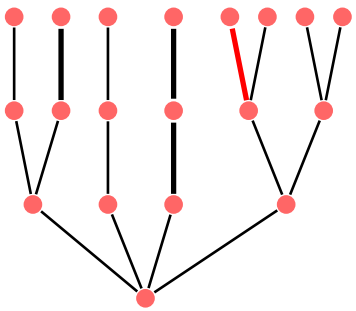
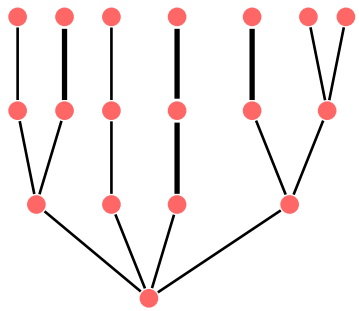
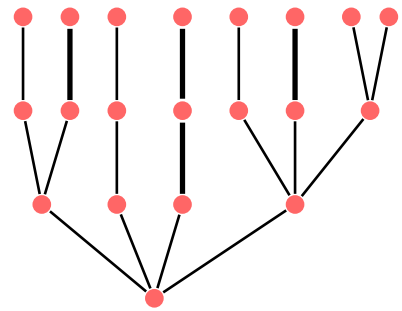
$$m_s + \sum_{i \le s} g(i) \le h\Big(\sum_{i<s} m_i\Big)$$

then h is an upper bound for the oracle use.

The choice $m_i = i$ is a nearly optimal choice giving oracle-use:

$$n + \sqrt{n} \cdot \log \sqrt{n}.$$

The code-tree is isomorphic to the full (m_i)-branching tree.

(ℓ_i)-labelable trees

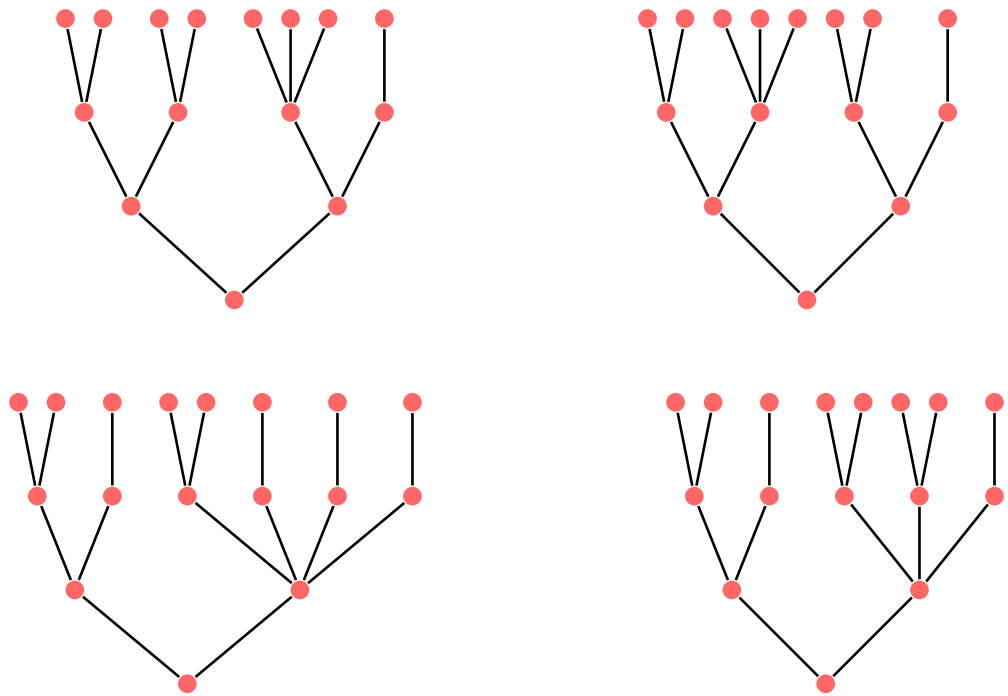
(1) consistency of string-labels

(2) level ℓ_n has at least one label for each $\sigma \in 2^n$

If P is a tree and $\mu(P) > \sum_{t=1}^{k} 2^{t-\ell_t}$ then it is (ℓ_i)–labelable up to level k.

Labelable iff it is splice-reducible to the full binary tree.

Not labelable



The new coding

(1) A greedy algorithm looking for the cheapest split;

(2) many identical labels co-exist at a level ℓ_{n}

(3) Each label has a unique active copy;

(4) the active copy of a label is the one placed last

(5) the inactive copies are saturated

(6) the weight of the active nodes is

$$\sum_{\mathrm{i}} 2^{\mathrm{i}-\ell_{\mathrm{i}}}$$

Theorem (Barmpalias and Lewis 2016)

Suppose that g is a nondecreasing computable function such that $\sum_i 2^{-g(i)} < 1$.

Then every sequence is computable from a random one with redundancy $g(n)$.

A new coding method into random sequences was needed....

...for any improvement of the known bounds.

If $\mathcal{P}$ is a Π^0_1 class and $\sum_i 2^{-g(i)} < \mu(\mathcal{P})$ then every real is computable from a member of $\mathcal{P}$ with redundancy g.

Why does it not terminate?

At any s, each labelled real in $\mathrm{P_s}$ has its largest label active.

If $\mathrm{D_s} = 2^\omega - \mathrm{P_s}$ and $\mathrm{U_s}(\nu)$ the set of active strings $\rho \supseteq \nu$,

For each s and labelled ν, we have $[\nu] \subseteq [\mathrm{D_s}] \cup [\mathrm{U_s}(\nu)]$

so if all extensions of λ become inactive,

$$1 \leq \Big(1 - \mu(\mathrm{P})\Big) + \sum_{\mathrm{i}} 2^{\mathrm{i}-\ell_{\mathrm{i}}}$$

contradiction.

Thank you for your attention!

Optimal redundancy in computations from random oracles *

George Barmpalias Andrew Lewis-Pye

Lower bounds on the redundancy in computations from random oracles via betting strategies with restricted wagers *

George Barmpalias Andrew Lewis-Pye Jason Teutsch