Bucharest, November 2009

Algorithmic Randomness: A Primer

Cristian S. Calude

UoA, Auckland, ENS, Paris

www.cs.auckland.ac.nz/~cristian

ENS, Paris, 20 October, 2009

Randomness

Probability theory, a calculus with "random objects", is silent about which individual objects are "random."

The simplest objects

1. finite bit-strings: cannot be "random" because they are finite,
2. infinite bit-sequences: cannot be "random" because of Ramsey theory (e.g. Green-Tao theorem: the sequence of primes contains arithmetic progressions of any length).

An algorithmic approach to randomness

The algorithmic approach to randomness consists in:

- adopting "incompressibility" as the basic symptom of randomness,
- measuring the quality of "algorithmic randomness" by the degree of "incompressibility".

Prefix-complexity

The *prefix-complexity* associated to a (prefix-free Turing) machine M (processing bit-strings into bit-strings) is defined by

$$H_M(x) = \inf\{|p| : M(p) = x\}.$$

Theorem. One can construct a *universal machine* U such that for every machine M there exists a constant $c = c_{U,M}$ with the property that for all strings x,

$$H_U(x) \leq H_M(x) + c.$$

Fix a universal machine U and write $H = H_U$.

String compressibility

From Leibniz's (1686) dictum: ▸ L

a theory must be simpler than the data is explains

one can arrive to the idea that

understanding is compression

which can be translated into a definition:

a string x is t–compressible if

$$H(x) \leq |x| - t,$$

meaning that $U(p) = x$ for some program p with $|p| \leq |x| - t$.

▸ Example

Algorithmic random strings

A string x is *(algorithmically) t–random* if $H(x) \geq |x| - t$, i.e. no program with less than $|x| - t$ bits can generate x via U.

Properties:

- for every t, t–random strings of every length exist;
- t–random strings satisfy all computable enumerable statistical properties of randomness;
- the set of t–random strings is immune;
- ZFC can prove t–randomness only for finitely many strings (none for some U);
- every "decent" Monte Carlo simulation algorithm (like Rabin's primality test) using t–random strings as samples produces the correct result (not only correct with high probability).

Prefix vs. time complexities

Theorem. There is a constant $c = c_U$ such that if a program p halts on U, then the time t_p it takes $U(p)$ to stop satisfies the inequality:

$$H(t_p) \leq |p| + c. \tag{1}$$

Can a program stop at an asymptotically random time?

A time t is "asymptotically random" if

$$H(t) \geq |t| - \log(|t|).$$

Theorem. Assume that $U(p)$ has not stopped ($|p| > 2$) by time $N = 2^{2|p|+2c+1}$, where c comes from (1). Then, $U(p)$ cannot exactly stop at any asymptotically random time $t \geq N$.

Theorem. Asymptotically random times are effectively dense.

An "anytime algorithm" for the halting problem

Theorem. For every positive integer N we can effectively compute a threshold time $\theta = \theta_{U,N}$ such that if a program of length N does not stop in time θ, then the set of times greater than θ at which the program can stop has effective zero density.

▸ ExactDensity

Corollary. Given a computable real $\varepsilon \in (0, 1)$ and positive integer N we can effectively compute a threshold time $\theta = \theta_{U,\varepsilon,N}$ such that the probability that a program of length N which has not stopped in time θ halts is smaller than ε.

▸ ProbabilitySpace

The above results are true for a large class of functions (including incomputable ones).

Complexity of infinite sequences

Consider a binary infinite sequence

$$\mathbf{x} = x_1 x_2 \cdots x_n \cdots$$

and its prefix of length n

$$\mathbf{x}(n) = x_1 x_2 \cdots x_n.$$

The prefix-complexity of $\mathbf{x}$ is given by the sequence

$$(H(\mathbf{x}(n))).$$

Reals are identified with their infinite binary expansions.

How complex is π?

Let

$$\mathbf{\Pi} = \pi_1 \pi_2 \cdots \pi_n \cdots$$

be the infinite binary sequence of the fractional part of π.

There is a constant $c > 0$ such that for each n,

$$H(\mathbf{\Pi}(n)) \leq 2 \log n + c.$$

How complex sequences can be?

Are there sequences $\mathbf{x}$ having all prefixes maximally incompressible:

$$\exists c \forall n \, [H(\mathbf{x}(n)) \geq n + H(n) - c]?$$

The answer is negative.

▸ MaxComplexity

Are there sequences $\mathbf{x}$ having all prefixes nearly random, i.e.

$$\exists c \forall n \, [H(\mathbf{x}(n)) \geq n - c]?$$

The answer is affirmative. They have maximum complexity and are called random.

How many reals are random?

The set of random reals

- has (constructive) Lebesgue measure one,
- is (constructively) meagre (Baire).

Omega numbers

The canonical example of random real is Chaitin's Omega, the "halting probability" of U:

$$\Omega_U = \sum_{U(p)\,\text{halts}} 2^{-|p|}.$$

With the first n bits of Omega we can solve the halting problem for every program of length less than or equal to n, so Omega achieves an exponential compression of the halting problem.

An Omega number is

- random,
- computably enumerable (as a real).

Theorem. Every computable enumerable random real is of the form Ω_U, for some universal machine U.

- (incompleteness) ZFC can prove the exact values of only finitely many digits of $\Omega_U = 0.\omega_1\omega_2\cdots$ (none for some Ω_U).

▸ EB

- Peano Arithmetic proves randomness of every c.e. random real (contrast: ZFC cannot prove randomness for more than finitely many strings).

Is quantum randomness random?

We don't know.

Theorem. Kochen-Specker theorem implies that an infinite sequence of quantum random bits is bi-immune. ▸ KS

Question. Is Rabin's probabilistic primality test powered with quantum random samples always correct?

THANK YOU!

References 1

1. M. Li, P. M. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications*, Springer, 3rd ed. 2009.
2. C. S. Calude. *Information and Randomness*, Springer, 2nd ed. 2002.
3. A. Nies. *Computability and Randomness*, Clarendon Press, 2009.
4. R. Downey, D. Hirschfeldt. *Algorithmic Randomness and Complexity*, Springer, 2010.

1. G. Chaitin, J.T. Schwartz. A note on Monte Carlo primality tests and algorithmic information theory *Comm. Pure Appl. Math.* 31 (1978), pp. 521-527.
2. C. S. Calude, M. A. Stay. Most programs stop quickly or never halt, *Adv. Appl. Math.* 40 (2008), 295–308.
3. Y. Manin. Renormalization and Computation II: Time Cutoff and the Halting Problem, `http://arxiv.org/abs/0908.3430`, 24 August 2009, 28 pp.

4. P. Martin-Löf. The definition of random sequences, *Inform. and Control* 9(1966), 602-619.
5. C. Calude, I. Chițescu. Random sequences: some topological and measure-theoretical properties, *An. Univ. București, Mat.-Inf.* 2 (1988), 27–32.
6. G. J. Chaitin. A theory of program size formally identical to information theory, *J. Assoc. Comput. Mach.* 22(1975), 329-340.
7. C. S. Calude, P. Hertling, B. Khoussainov, Y. Wang. Recursively enumerable reals and Chaitin Ω numbers, *Theoret. Comput. Sci.* 255 (2001), 125–149.
8. A. Kučera, T. A. Slaman. Randomness and recursive enumerability, *SIAM J. Comput.* 31, 1 (2001), 199–211.

9. R. M. Solovay. A version of Ω for which *ZFC* can not predict a single bit, Calude, Păun (eds.). *Finite Versus Infinite. Contributions to an Eternal Dilemma*, Springer, 2000, 323–334.
10. C. S. Calude. Chaitin Ω numbers, Solovay machines and incompleteness, *Theoret. Comput. Sci.* 284 (2002), 269–277.
11. C. S. Calude, M. J. Dinneen. Exact approximations of omega numbers, *Int. J. Bif. & Chaos* 17, 6 (2007), 1937–1954.
12. C. S. Calude, Elena Calude, M. J. Dinneen. A new measure of the difficulty of problems, *J. Multiple-Valued Logic Soft Comput.* 12 (2006), 285–307.
13. C. S. Calude, N. Hay. Every computably enumerable random real is provably computably enumerable random, *Logic Journal IGPL*, June, 2009.
14. C. S. Calude, K. Svozil. Quantum randomness and value indefiniteness, *Advanced Science Letters* 1 (2008), 165–168.

Leibniz

Sans les mathématiques on ne pénètre point au fond de la philosophie.
Sans la philosophie on ne pénètre point au fond des mathématiques.
Sans les deux on ne pénètre point au fond de rien.

▸ String Compressibility

Is this text compressible?

Study this paragraph and all things in it. What is vitally distinct about it? Actually, nothing is wrong, but you must admit that it is most unusual. Don't just zip through it quickly but study it scrupulously. With a bit of luck you should spot what it is so particular about it and all words found in it. Can you say what it is? Try hard as isn't it all that difficult.

▸ String Compressibility

If an N-bit program runs for time $T > \max\{\theta_N, 2^{2+5\cdot 2^k}\}$, then the density of times at which the program can stop is less than 2^{-k}.

▸ Density

We now consider the probability space to be

$$Space_{\{\rho(i)\}} = \Sigma^* \times \{1, 2, \ldots\},$$

where N-bit programs are assumed to be uniformly distributed, and the runtime is given by the computable probability distribution $\{\rho(i)\}$.

▸ Density

Leibniz

$$\max_{|x|=n} H(x) = n + H(n) + \mathrm{O}(n).$$

▸ ComplexSeq

A glimpse of an Omega

The first exact 40 bits of a natural Omega are:

00010 00000 01000 01010 01110 11100 00111 11010

To solve the

- Riemann's hypothesis we need the first 4,680 bits
- the four colour theorem we need the first 4,336 bits
- Goldbach's conjecture we need the first 756 bits

of the same Omega.

▸ Paradox

(Value definiteness: **VD**) All observables defined for a QM system have definite values at all times.

(Noncontextuality: **NC**) If a QM system possesses a property (value of an observable), then it does so independently of any measurement context, i.e. independently of how that value is eventually measured.

Kochen-Specker theorem: In QM, **VD** + **NC** is contradictory. For example, given **NC** certain sets of QM observables cannot consistently be assigned values at all. ▸ QR

Let H be a Hilbert space of QM state vectors of dimension $n \geq 3$. There is a set M of m observables on H such that the assumptions (KS1) and (KS2) are contradictory:

(KS1) Every element A of M has a value $v(A)$.

(KS2) Values of observables satisfy the rules:
(a) If A, B, C are all compatible and $C = A + B$, then $v(C) = v(A) + v(B)$;
(b) if A, B, C are all compatible and $C = AB$, then $v(C) = v(A)v(B)$.

Example: $n = 4$ and $m = 18$. ▸ QR