Randomness is best understood through various "symptoms". Here are three of the largely accepted ones:

(i) *Unpredictability:* The impossibility of any agent, acting via uniform, effective means, to predict correctly and reproducibly the outcome of an experiment using finite information extracted from the environment.

(ii) *Incompressibility:* The impossibility to compress a random sequence.

(iii) *Typicality:* Random sequences pass every statistical test of randomness.

The French mathematician Émile Borel, a pioneer of probability theory, argued that there is no way to formalise in an acceptable way the concept of randomness. His argument is based only one symptom of randomness, typicality, and is known as the *randomness paradox*.

A random bit-string should be "typical": it should not stand out from the crowd of other bit-strings.

Here is Borel's argument. Can we have a precise way to distinguish between "random bit-strings" and bit-strings which are "non-random"? It does not matter how this criterion was found or operates.

Assume we have a precise **criterion** which can be applied to any bit-string and once a bit-string is given, we can say whether the bit-string is *random* or *non-random*.

Can the **criterion** be consistent? The answer is **negative**.

Indeed, choose the first bit-string which **criterion** asserts it is random. This particular bit-string is

the first bit-string satisfying the property of being random,

a property making it atypical, so non-random!

A prefix-free (or self-delimiting) program is a program whose domain is prefix-free.

- The identity $\psi(x) = x$, $x \in B^*$ is not computable by a prefix-free program; no total function is computable by a prefix-free program.
- The restriction of the identity $\psi(x) = x$ to the set $\{1^n 0 \mid n \geq 1\}$ is computable by a prefix-free program.
- The function $\theta(1^{|x|}0x) = x$, for all $x \in B^*$ is computable by a prefix-free program.
- The set of prefix-free programs is computably enumerable.

UTheorem

We can effectively construct a program W such that for every program T there effectively exists a constant $a = a_{W,T}$ such that for each input string x there exists an input y with $|y| \leq |x| + a$ such that $W(y) = T(x)$.

UPFTheorem

We can effectively construct a prefix-free program U such that for every prefix-free program C there effectively exists a constant $c = c_{U,C}$ such that for each input string x there exists an input z with $|z| \leq |x| + c$ such that $U(z) = C(x)$.

▸ Proof

Let M be a prefix-free program. The program-size complexity $H_M(x)$ is the size in bits of the smallest program for M to compute x:

$$H_M(x) = \inf\{|p| \mid p \in B^*, M(p) = x\}.$$

When $M = U$ is universal, then H_U is simply denoted by H.

The program-size of the sequence $\mathbf{x}$ is given by the sequence

$$(H(\mathbf{x}(n))_{n>0}.$$

Challenges

1. One can effectively construct a prefix-free program U such that for every prefix-free program C there effectively exists a constant $c = c_{U,C}$ such that for each input string $x, H_U(x) \leq H_C(x) + c$.

2. Prove that for every universal prefix-free programs U and U' there is a constant $c = c_{U,U'}$ such that for all strings x:

$$\mid H_U(x) - H_{U'}(x) \mid \leq c.$$

Examples

- *low complexity strings:*

$$H(0^n) \approx \log n + c, H(1^n) \approx \log n + c$$

- *intermediate complexity strings (borderline algorithmically random strings):* if $x^* = \min\{p \mid U(p) = x\}$, then

$$H(x) = |x^*|, H(x^*) \geq |x^*| - c$$

- *high complexity strings (algorithmically random strings):*

$$\max\{H(x) \mid |x| = n\} = n + H(\mathrm{bin}(n)) + c \approx n + 2\log n + c$$

▸ Proof

Challenges

1. The program-size complexity H is not computable.
2. The set $\{x^* \mid x \in B^*\}$ is immune, that is, it contains no infinite c.e. subset.
3. For every $n > 0$ there exists $x \in B^*$ such that $|x| =$ and $H(x) \geq |x|$.

Example

The binary representation of the number 0.40626 is

$$0.011010000\cdots$$

The complexity of the sequence

$$\mathbf{x} = 011010000\cdots$$

is bounded by a constant, i.e. there exists $c > 0$ such that for each n,

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

Example

Consider the number π written in binary and let

$$0.\pi_1\pi_2\cdots\pi_n\cdots$$

be the infinite binary sequence of the fractional part of π. There is a constant $c' > 0$ such that for each n,

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

where

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

As the Halting Problem is undecidable, we can approach the problem probabilistically.

Chaitin's Omega is the "halting probability" of U is:

$$\Omega_U = \sum_{p \in \mathrm{dom}(U)} 2^{-|p|}.$$

Challenge

Prove that for every prefix-free set $S \subset B^*$,

$$\sum_{p \in S} 2^{-|p|} \leq 1,$$

and deduce that

$$\Omega_U \in (0, 1).$$

Theorem [Omega solves the halting problem]

With the first n bits of Omega we solve the halting problem for every program of length less than or equal to n.

▸ Proof

Corollary [Incomputability of Omega]

The sequence $\Omega_U = \omega_1\omega_2\cdots\omega_n\cdots$ is not computable.

Proof hint: Is $\mathrm{dom}(U)$ computable?

Some of these bits have actually been determined. For a natural choice of the program U, the first 40 bits of $\Omega = \Omega_U$ are

$$0001000000010000101001110111000011111010,$$

see [11].

If we knew the first 5,000 bits, we would know if the Riemann hypothesis is correct. For the Four colour theorem we need even less bits. ▸ OmegaMedia

Corollary [Transcendence of Omega]

The number Ω is transcendental and $\Omega \in (0, 1)$.

Theorem [Bi-immunity of Omega]

The sequence $\omega_1\omega_2\cdots\omega_n\cdots$ is bi-immune, that is, no algorithm can compute exactly more than finitely many bits of ω_i.

See proofs in [8, 12].

Theorem [Omega has maximum complexity]

If

$$\Omega_U = 0.\omega_1\omega_2\cdots\omega_n\cdots$$

then there exits a constant $c > 0$ such that for all $n \geq 1$,

$$H(\omega_1\omega_2\cdots\omega_n) \geq n - c.$$

Reals whose binary expansions are sequences with maximum complexity are called algorithmically random, shortly random.

▸ Proof

Theorem [Omega is c.e.]

The number Omega is c.e., that is, the limit of an increasing computable sequence of rationals in $(0, 1)$.

▸ Proof

So, Omega is simultaneously c.e. and algorithmically random.

In fact, the converse is also true:

Theorem [Omega = c.e. and random]

The set of Omega numbers coincides with the set of c.e. random numbers.

See more in [8, 14].

A sequence $\mathbf{x}$ is Martin-Löf random iff there exits a constant $c > 0$ such that for all $n \geq 1$,

$$H(\mathbf{x}(n)) \geq n - c.$$

See more in [8, 14].

Corollary

Every Omega is Martin-Löf random.

Corollary

Every pseudo-random sequence is not Martin-Löf random.

Is Martin-Löf randomness a good model of randomness?

Martin-Löf randomness

- is provably better than pseudo-randomness,
- satisfies the typicality requirement,
- satisfies the incompressibility requirement,
- is believed to satisfy the requirement of unpredictability, but not convincing model was yet developed to test it,
- no known device for producing it has been yet designed and certified.

See more in [8, 14].

There are many random number generators:

1. pseudo-random generators, software produced
   - PCG, Random123, xoroshilro128+
2. hardware generators, devices that generate random numbers from physical processes
   - macroscopic, e.g. coin, dice, roulette wheels, lottery machines,
   - microscopic, e.g. thermal noise, photoelectric effect, quantum effects.

In particular, there are many quantum random generators, from lab experiments to openly accessible on the internet and commercial like Quantis.

According to NIST, New quantum method generates really random numbers, phys.org/news/, [19, 7]:

"Quantum mechanics provides a superior source of randomness because measurements of some quantum particles (those in a "superposition" of both 0 and 1 at the same time) have fundamentally unpredictable results."

But,

- what does it mean superior?
- is the answer "because measurements of some quantum particles have fundamentally unpredictable results" good enough?

Why do we need another quantum random generator?

Because no current quantum random generator (QRNG) is provably better than the other generators, in particular, pseudo-random generators.

Is this so? Many advantages promised by QRNGs rely on the belief that the outcomes of quantum measurements are

intrinsically/irreducibly unpredictable.

This belief underlies:

- the use of QRNGs to produce "quantum random" sequences that are "truly unpredictable",
- the generation of cryptographic keys unpredictable to any adversary.

Is this belief reasonable?

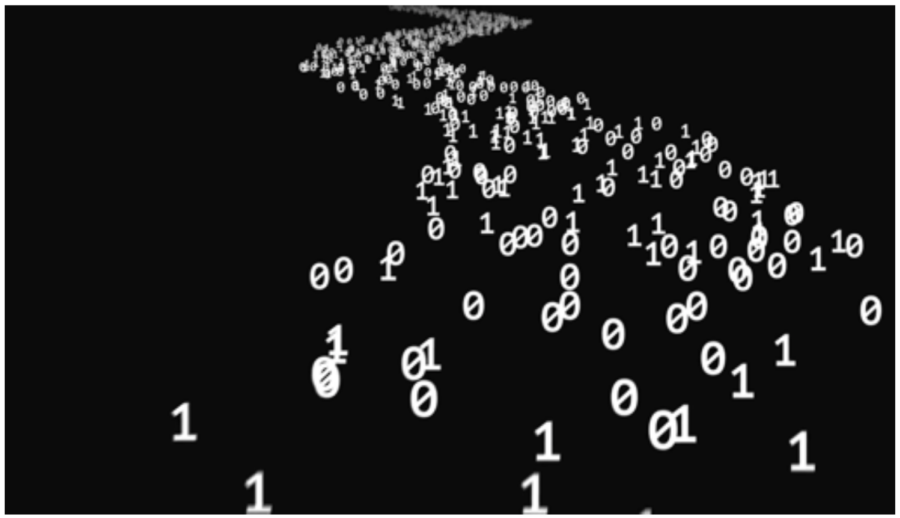
Work reported in *Nature* (doi:10.1038/news.2010.181, 14 April 2010) claims that quantum randomness is true randomness:

True randomness

More recently, an article published in July 2019 in Wired magazine claims that quantum computers could produce true randomness:

ANIL ANANTHASWAMY SCIENCE 06.23.19 08:00 AM

QUANTUM COMPUTERS COULD BE TRUE RANDOMNESS GENERATORS



The goal is an unending stream of perfectly random 1s and 0s. DAVE WHYTE/QUANTA MAGAZINE

SAY THE WORDS "quantum supremacy" at a gathering of computer scientists, and eyes will likely roll. The phrase refers to the idea that quantum computers will soon cross a threshold where they'll perform with relative ease tasks that are extremely hard for classical computers. Until recently, these tasks were thought to have little real-world use, hence the eye rolls.

If "true randomness" means "lack of any correlations" or "maximal randomness", does it exist?

Challenge: What is the answer? Proof?

Questions

- How can we produce quantum randomness?
- What is the physical "reason" of quantum randomness?
- What is the quality of quantum randomness?
- Is quantum randomness better than pseudo-randomness?

- Quantum randomness is
  - postulated and
  - generally reduced to the indeterminism of quantum measurements: because the outcome is indeterministic there is no way to predict it, hence it is random

- However, indeterminism does not imply randomness and randomness does not imply indeterminism:
  - pseudo-randomness
  - coin-tossing (chaoticity)
  - Schrödinger equation

Is quantum randomness provable better than pseudo-randomness?

How should we formulate mathematically the question in the title?
Challenge: What is the answer?

Classical computers use bits which are physically represented by two-state classical systems (two positions of an electrical switch, two distinct voltage or current levels allowed by a circuit).

Quantum computers operate with qubits (quantum bits) which are described as quantum states (i.e. abstract, in Hilbert space) that are physically implemented by quantum systems (e.g. an atom) which are in one of two definite states.

For example, the state of a spin-$\frac{1}{2}$ particle, when measured, is always found to be in one of two possible states, represented – using Dirac bra-ket notation – as

$$|+\frac{1}{2}\rangle \text{ (spin-up) or } |-\frac{1}{2}\rangle \text{ (spin-down).}$$

Formally, a qubit is denoted as

$$|0\rangle \text{ or } |1\rangle.$$

Unlike the intermediate states of a classical bit (e.g. any voltages between the "standard" representations of 0 and 1) which can be distinguished from 0 and 1, but do not exist from an informational point of view, quantum intermediate states cannot be reliably distinguished, even in principle, from the basis states, but do have an informational "existence".

A superposition state $|\varphi\rangle$ (pronounced "ket phi") is a qubit state vector represented by a linear combination of basis states conventionally denoted by $|0\rangle$ and $|1\rangle$, that is

$$|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle,$$

where α, β are complex numbers with $\alpha^2 + \beta^2 = 1$.

A qubit is a unit vector in the space $\mathbf{C}^2$, so for each qubit $|x\rangle$, there are two (complex) numbers $a, b \in \mathbf{C}$ such that

$$|x\rangle = a|0\rangle + b|1\rangle = \begin{pmatrix} a \\ b \end{pmatrix}, \tag{1}$$

where

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \; |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

and $|a|^2 + |b|^2 = 1$.

We can perform a measurement that projects the qubit onto the basis $\{|0\rangle, |1\rangle\}$. Then we will obtain the outcome $|1\rangle$ with probability $|b|^2$, and the outcome $|0\rangle$ with probability $|a|^2$.

With the exception of limit cases $a = 0$ and $b = 0$, the measurement irrevocably disturbs the state:

> *If the value of the qubit is initially unknown, then there is no way to determine a and b with any conceivable measurement.*

However, *after* performing the measurement, the qubit has been prepared in a known state (either $|0\rangle$ or $|1\rangle$); this state is typically different from the previous state.

The ability of quantum systems to exist in a "blend" of all their allowed states simultaneously is known as the Principle of Superposition.

Even though a qubit can be put in a superposition (1), it contains no more information than a classical bit, in spite of its having infinitely many states. The reason is that information can be extracted only by measurement. But, as we have argued, for any measurement of a qubit with respect to a given orthonormal basis, there are only two possible results, corresponding to the two vectors of the basis.

Non cloning

It is not possible to capture more information measuring in two different bases because the measurement changes the state. Even worse, quantum states cannot be cloned, hence it's impossible to measure a qubit in two different ways (even, indirectly, by using a copy trick, that is copying and measuring the copy).

Let's measure the qubits

$$\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \text{ and } \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle.$$

Do we get a random bit? Yes, because the probability of getting 0 is equal to the probability of getting 1:

$$\left(\frac{1}{\sqrt{2}}\right)^2 = \frac{1}{2}.$$

Is this true randomness?

How random are the outcomes?

Value indefiniteness is a central notion to this discussion. Informally, for a given quantum system in a particular state, we say that an observable is value definite if the measurement of that observable is predetermined to take a (potentially hidden) value.

In defining it we are guided by Einstein, Podolsky and Rosen famous analysis in [15] which can be encapsulated in the following

> *EPR principle: If, without in any way disturbing a system, we can predict with certainty the value of a physical quantity, then there exists a definite value prior to observation corresponding to this physical quantity.*

See more in [1, 3].

EPR principle justifies also

Eigenstate principle: *If a quantum system is prepared in a state $|\psi\rangle$, then the projection observable P_ψ is value definite.*

- Admissibility: Definite values must not contradict the statistical quantum predictions for compatible observables on a single quantum.
- Noncontextuality of definite values: The outcome obtained by measuring a value definite observable (a preexisting physical property) is *noncontextual*, i.e. it does not depend on other compatible observables which may be measured alongside the value definite observable.
- Eigenstate principle: If a quantum system is prepared in the state $|\psi\rangle$, then the projection observable P_ψ is value definite.

Assume the above three hypotheses.

Theorem [1]

Assume a quantum system prepared in the state $|\psi\rangle$ in dimension $n \geq 3$ Hilbert space $\mathfrak{C}^n$, and let $|\phi\rangle$ be any state neither orthogonal nor parallel to $|\psi\rangle$. Then the projection observable P_ϕ is value indefinite.

> *EPR principle:* *If, without in any way disturbing a system, we can predict with certainty the value of a physical quantity, then there exists a definite value prior to observation corresponding to this physical quantity.*

Conversely, if no unique element of physical reality corresponding to a particular physical quantity exists, this is reflected by the physical quantity being value indefinite.

If a physical property is value indefinite we cannot predict with certainty the outcome of any experiment measuring this property.

If we assert of an observable event that it is unpredictable we do not mean, of course, that it is logically or physically impossible for anybody to give a correct description of the event in question before it has occurred; for it is clearly not impossible that somebody may hit upon such a description accidentally. What is asserted is that certain rational methods of prediction break down in certain cases—the methods of prediction which are practised in physical science.

We present a non-probabilistic model of prediction based on the ability of a computable operating agent to correctly predict using finite information extracted from the system of the specified experiment, [4].

Predictions should remain correct in any arbitrarily long (but finite) set of repetitions of the experiment.

Consider a physical experiment E producing a single bit.
We consider an experiment E producing a single bit $x \in \{0, 1\}$.

An example of such an experiment is the measurement of a photon's polarisation after it has passed through a 50-50 beam splitter.

With a particular trial of E we associate the parameter λ (the state of the universe) which fully describes the trial. While λ is not in its entirety an obtainable quantity, we can view it as a resource that one can extract finite information from in order to predict the outcome of the experiment E.

An extractor is a (deterministic) function $\lambda \mapsto \langle\lambda\rangle$ mapping reals to rationals.

For example, $\langle\lambda\rangle$ may be an encoding of the result of the previous trial of E, or the time of day the experiment is performed.

A predictor for E is an algorithm (computable function) P_E which halts on every input and outputs **0** or **1** or **prediction withheld**.

P_E can utilise as input the information $\langle\lambda\rangle$, but, *as required by* EPR, must be passive, i.e. must not disturb or interact with E in any way.

A predictor P_E provides a correct prediction using the extractor $\langle\,\rangle$ for a trial of E with parameter λ if, when taking as input $\langle\lambda\rangle$, it outputs $\mathbf{0}$ or $\mathbf{1}$ and this output is equal to the result of the experiment.

The predictor P_E is $k, \langle\,\rangle$-correct if there exists an $n \geq k$ such that when E is repeated n times with associated parameters $\lambda_1, \ldots, \lambda_n$ producing the outputs $x_1, x_2, \ldots, x_n$, P_E outputs the sequence $P_E(\langle\lambda_1\rangle), P_E(\langle\lambda_2\rangle), \ldots, P_E(\langle\lambda_n\rangle)$ with the following two properties:

(i) no prediction in the sequence is incorrect, and

(ii) in the sequence there are k correct predictions.

The outcome x of a single trial of the experiment E performed with parameter λ is predictable (with certainty) if there exist an extractor $\langle\,\rangle$ and a predictor P_E which is

1. $k, \langle\,\rangle$-correct for all k, and
2. $P_E(\langle\lambda\rangle) = x$.

Theorem 3

If E is an experiment measuring a quantum value indefinite observable, then every predictor P_E using any extractor $\langle\,\rangle$ is not $k, \langle\,\rangle$-correct, for all k.

Theorem 4

In an infinite repetition of E as considered above, no single bit x_i of the generating infinite sequence $x_1 x_2 \ldots$ can be predicted.

> **epr principle**: *If a repetition of measurements of an observable generates a computable sequence, then this implies these observables were value definite.*

Theorem 5 [1]

Assume the epr and Eigenstate principles. An infinite repetition of the experiment E measuring a quantum value indefinite observable generates an incomputable (even bi-immune, i.e. no algorithm can compute correctly more than finite many digits of the) infinite sequence $x_1 x_2 \ldots$.

Generalised beam QRNG producing maximum unpredictable, incomputable sequences

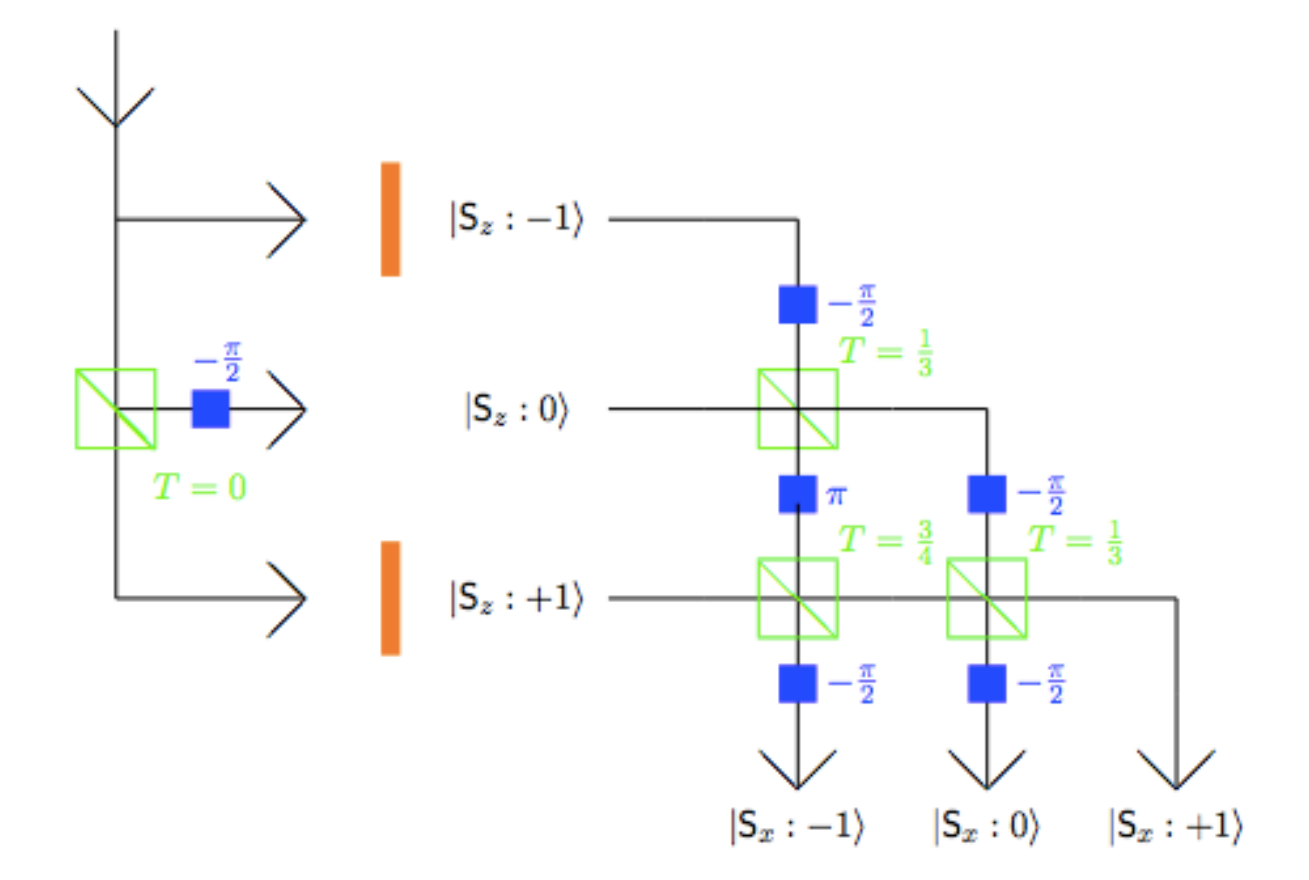


Figure: Generalised beam QRNG

Realisation of the generalised beam 3D QRNG with qutrits

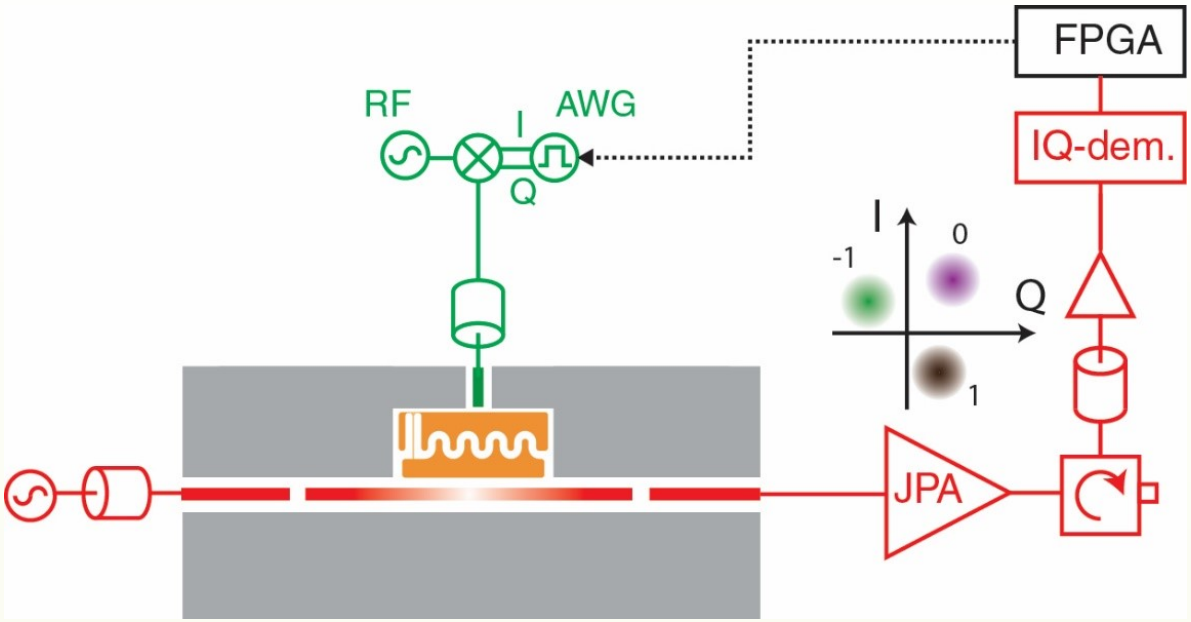


Figure: Realisation of a QRNG with superconducting circuits on a chip (gray). A qutrit (yellow) is coupled to a read out (red) and control circuitries. Near quantum limited Josephson parametric amplifier (JPA) can be used to discriminate all possible outcomes of the protocol with certainty including false runs. Fast programmable gate array (FPGA) can be used to provide the correction pulses to reinitialise the qutrit in its ground state right after the end of the protocol run. [17]

The theory developed above guarantees the bi-immunity and unpredictability of every sequence produced by the generalised beam 3D QRNG with qutrits.

The experimental analysis of the quantum random bits [2] reported only weak evidence of incomputability. Some possible reasons are;

1. a problematic branch with probability zero used in the generalised beam splitter,
2. not long enough length of samples,
3. imperfections in the implementation of the measuring protocol.

An new 3D QRNG with a stronger theoretical certification was developed in [6].

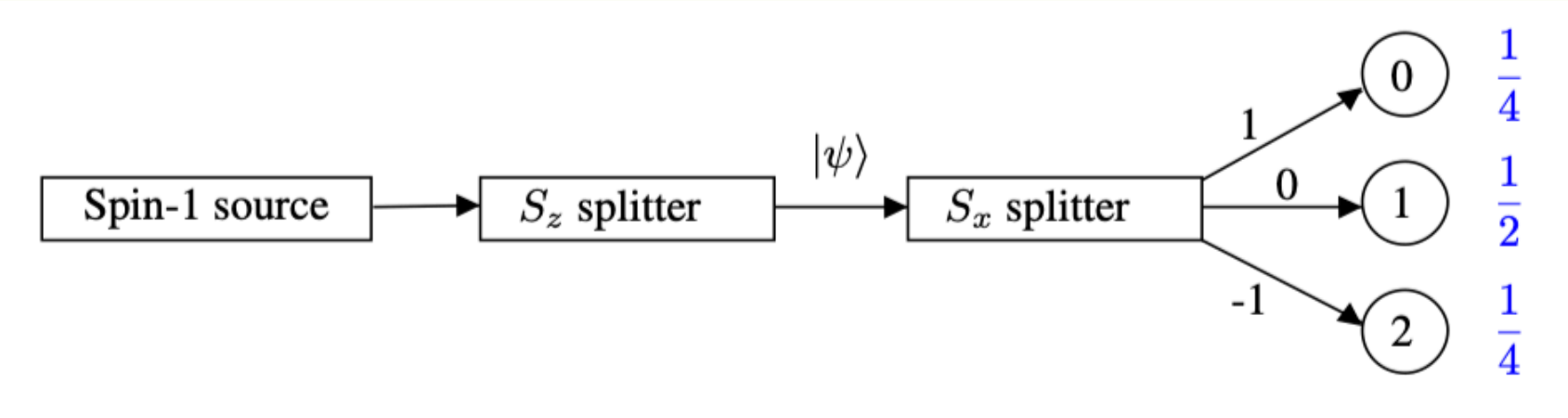


Figure: 3D QRNG with probabilities $1/4, 1/2, 1/4$.

As many applications require binary random strings, the following computable alphabetic morphism $\varphi : \{0, 1, 2\} \to \{0, 1\}$

$$\varphi(a) = \begin{cases} 0, & \text{if } a = 0, \\ 1, & \text{if } a = 1, \\ 0, & \text{if } a = 2, \end{cases}$$

transforms by sequential concatenation ternary strings/sequences into binary ones and preserves the validity of Theorems 3,4, 5.

A primality test is an algorithm for determining whether an input number is prime.

Primality tests are used in many fields, including cryptography.

Primality is considered computationally *easy* because polynomial algorithms in the size of the input solve it; the first was in 2004 [5].

In contrast, factorization is thought, but not proved, to be a computationally *difficult* problem.

So, there is nothing to talk about primality testing...

Well, not!

The polynomial algorithms for primality – currently the best runs in time $O((\log n)^6)$ – are practically inefficient for large numbers.

Hence these algorithms are not useful for applications requiring large primes; instead, efficient probabilistic algorithms are used.

A randomized algorithm is an algorithm that uses a degree of randomness to enhance computation.

Frequently, such an algorithm uses uniformly distributed random bits in the hope of achieving good performance in the "average case".

A probabilistic primality test works like this:

1. Given N, "randomly" pick a positive integer $0 < a < n$.
2. Fast check a predicate depending on n and a.
3. If the predicate is true, then n is composite and a is a witness for the compositeness n.
4. Otherwise, re-do steps 2 and 3 till a witness for the (certainly) compositeness is found, or the *required accuracy is reached*. In the last case, n is declared probably prime.

Solovay-Strassen test [21, 23] checks the primality of a positive integer n as follows:

Take k natural numbers uniformly distributed between 1 and $n-1$, inclusive, and, for each $i = i_1, \ldots, i_k$, check whether a certain, easy to compute (Solovay-Strassen) predicate $W(i, n)$ holds:

1. If $W(i, n)$ is true then "i is a witness of n's compositeness", hence n is composite.
2. If $W(i, n)$ holds for at least one i then n is composite; otherwise, the test is inconclusive, but in this case the probability that n is prime is greater than $1 - 2^{-k}$.

This is due to the fact that *at least half* the i's between 1 and $n-1$ satisfy $W(i, n)$ if n is composite, and *none* of them satisfy $W(i, n)$ if n is prime [22].

This test runs in time $O(k \cdot \log_3 n)$. It has shown the practical feasibility of the RSA cryptosystem.

Research Challenge

Can we improve the quality of the algorithm, ideally removing the inconclusive verdict? More precisely, can we transform the probabilistic algorithm into a deterministic one?

Consider $s = s_0 \dots s_{m-1}$ a binary string (of length m) and n an integer greater than 2. Let k be the smallest integer such that $(n-1)^{k+1} > 2^m - 1$; we can thus rewrite the number whose binary representation is s into base $n-1$ and obtain the unique string $d_k d_{k-1} \dots d_0$ over the alphabet $\{0, 1, \dots, n-2\}$, that is,

$$\sum_{i=0}^{k} d_i (n-1)^i = \sum_{t=0}^{m-1} s_t 2^t.$$

The predicate $Z(s, n)$ is defined by

$$Z(s, J) = \neg W(1 + d_0, n) \wedge \dots \wedge \neg W(1 + d_{k-1}, n), \qquad (2)$$

where W is the Solovay-Strassen predicate.

> **Theorem** *For all sufficiently large c, if s is a c-incompressible string of length $\ell(\ell+2c)$ and n is an integer whose binary representation is ℓ bits long, then $Z(s,n)$ is true if and only if n is prime.*

The crucial fact is that the set of c-incompressible binary strings is highly incomputable: technically the set is immune, that is, it contains no infinite computably enumerable subset [9]. As a consequence, de-randomisation is thus *non-constructive*, and thus without practical value.

Has Chaitin-Schwartz Theorem no practical value? **No!** We can transform it into a test of incomputability [10]!

To this aim we use Carmichael numbers as these target composites.

A Carmichael number is a composite positive integer n satisfying the congruence $b^{n-1} \equiv 1 \pmod{n}$ for all integers b relatively prime to n. Although Carmichael numbers are composite, they are difficult to factorise and thus are "very similar" to primes; they are sometimes called pseudo-primes.

Many Carmichael numbers can pass Fermat's primality test [24], but less of them pass the Solovay-Strassen test. Increasingly Carmichael numbers become "rare": there are 1,401,644 Carmichael numbers in the interval $[1, 10^{18}]$.

In order to carry out the test we first fix c. For each Carmichael number n (with an ℓ-bit binary representation) we take $c = \ell - 1$. This choice is somewhat arbitrary; other choices could of course be made but would make little difference to our test.

For each n we take $\ell\,(\ell + 2c)$ bits.
Rewriting s in base $n-1$ as above, we then compute $W(1 + d_j, n)$ for $0 \leq j \leq k$ until the first j is found such that $W(1 + d_j, n)$ holds (and the compositeness of n is thus witnessed).

The test looks for direct violations of the Chaitin-Schwartz Theorem: a violation appears when for all $j = 0, \ldots, k-1$, $W(1 + d_j, n)$ are false; that is, all tests wrongly conclude that n is "probably prime".

As the Solovay-Strassen test guarantees that $W(1 + d_j, n)$ is true with probability at least one-half when n is a composite number, it quickly becomes difficult, in practice, to observe such violations for even the smallest Carmichael numbers used in the previous tests.

In order to observe some violations we restrict the test to the odd composite numbers less than 50: $9, 15, 21, 25, 27, 33, 35, 39, 45, 49$. For these numbers, we compute $Z(s, n)$ by reading $\ell(\ell + 2c)$ bits and following the same procedure as in the third test. When $Z(s, n) = 1$, a violation of the Chaitin-Schwartz Theorem is thus observed.

Since testing this predicate a single time on the ten numbers above would give insufficient statistics to observe any difference between the sources, we then repeated the above procedure reading from the 2nd bit of each string, then the 3rd, etc., until all the random bits have been used.

The metric is thereby taken as the average number of violations observed for the 10 composites tested (where the average is taken over all the repetitions), [2].

Results of testing incomputability

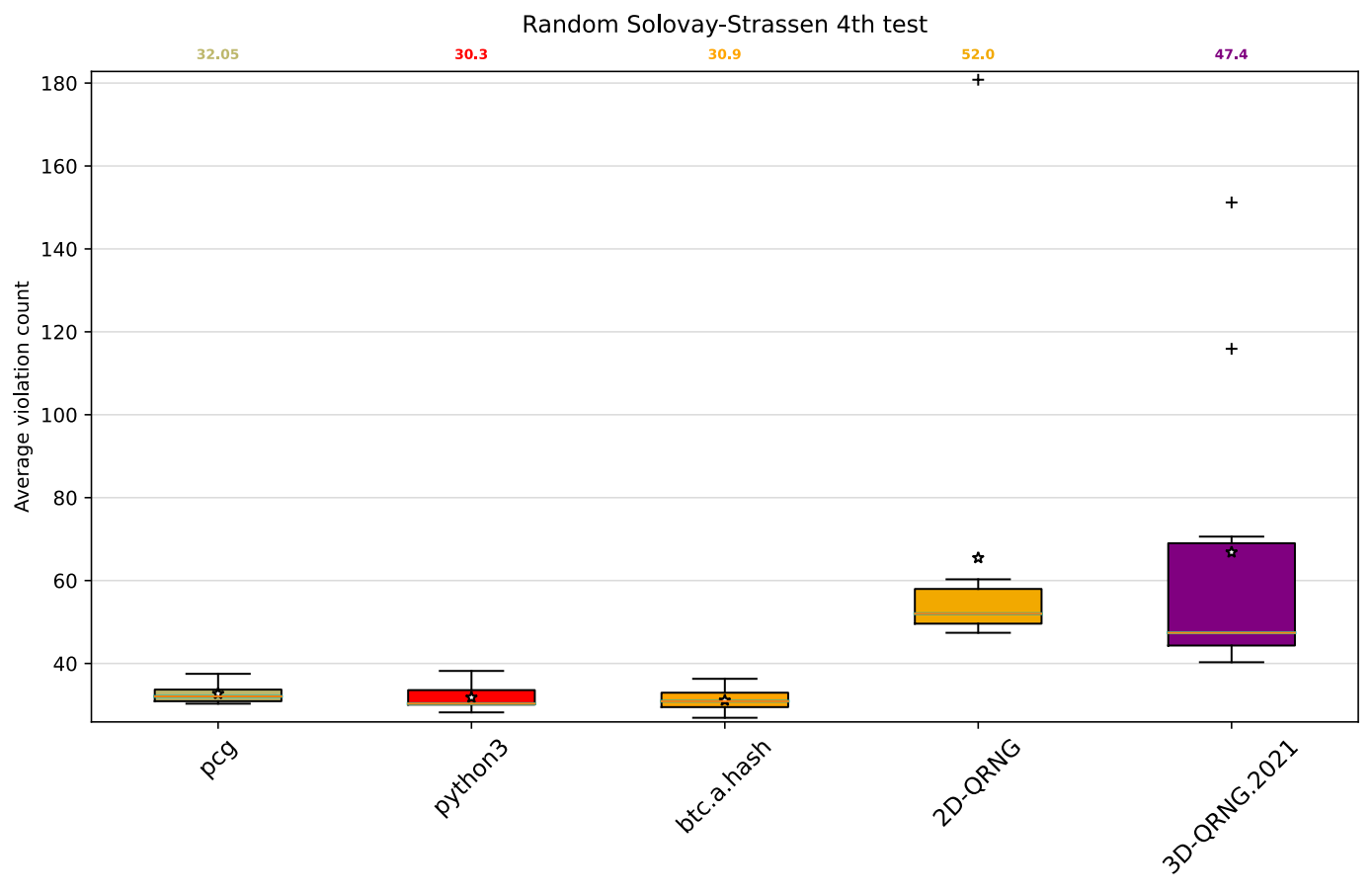


Figure: The table shows the results of testing 10 strings of length 2^{32} from each sample.

The results of the Kolmogorov-Smirnov tests [13] which there are shows statistically significant differences between the probability distributions of PRNGs outcomes and QRNGs ones.

Are the main assumptions, Admissibility, Non-contextuality, Eigenstate and epr principles, used in the proof of Theorems 3,4,5, physically "acceptable"?

A cautiously affirmative answer to this question comes from the results of the testing incomputability of strings of length 2^{32} with Chaitin-Schwartz Theorem.

Proofs

Proof

Let C_i be a computably enumeration of all prefix-free programs and construct the prefix-free program U by

$$U(1^i 0x) = C_i(x).$$

▸ USTheorem

Proof

To prove that $H(x^*) \geq |x^*| - c$ we construct the prefix-free program

$$D(p) = U(U(p))$$

and pick the constant c coming from the universality theorem (applied to U and D). Take $x = y^*$, $z = x^*$. One has:

$$D(z) = U(U(z)) = U(U(x^*)) = U(x) = U(y^*) = y,$$

so

$$H_D(y) \leq |z| = |x^*| = H(x),$$

$$|x| = |y^*| = H(y) \leq H_D(y) + c \leq H(x) + c.$$

▸ StringComplexity

Proof

Given $\omega_1\omega_2\cdots\omega_n$ we run in parallel $U(p)$ on various programs till we get enough programs $p_1, \ldots, p_N$ such that

$$\sum_{i=1}^{N} 2^{-|p_i|} \geq 0.\omega_1\omega_2\cdots\omega_n.$$

Every program q with $|q| \leq n$, different from all p_i's above, doesn't stop as otherwise

$$\Omega < 0.\omega_1\omega_2\cdots\omega_n + 2^{-n} \leq \sum_{i=1}^{N} 2^{-p_i} + 2^{-|q|} \leq \Omega,$$

a contradiction. ▸ OmegaHaltTheorem

Proof

Define M as follows: on input x compute $y = U(x)$ and the smallest number of programs $p_1, \ldots, p_t \in \mathrm{dom}(U)$ (if they exist) with

$$\sum_{i=1}^{t} 2^{-|p_i|} \geq 0.y.$$

Let $M(x)$ be the first (in quasi-lexicographical order) string not belonging to the set $\{U(p_1), U(p_2), \ldots, U(p_t)\}$ if both y and t exist, and $M(x) = \infty$ if $U(x) = \infty$ or t does not exist.

If $M(x) < \infty$ and x' is a string with $U(x) = U(x')$, then $M(x) = M(x')$.

Applying this to an arbitrary x with $M(x) < \infty$ and the program $x' = (U(x))^*$ yields

$$H_M(M(x)) \leq |x'| = H_U(U(x)). \tag{3}$$

Furthermore, by the universality of U there is a constant $c > 0$ with

$$H_U(M(x)) \leq H_M(M(x)) + c, \tag{4}$$

for all x with $M(x) < \infty$.

Take $U(x) = \omega_1 \cdots \omega_n$. From the construction of M (and the fact that Omega solves the halting problem) we conclude that $H_U(M(x)) \geq n$. Using (4) and (3) we obtain

$$\begin{aligned} n &\leq H_U(M(x)) \\ &\leq H_M(M(x)) + c \\ &\leq H_U(U(x)) + c \\ &= H_U(\omega_1\omega_2 \cdots \omega_n) + c. \end{aligned}$$

▸ OmegaMaxComplTheorem

Proof

The sequence of rationals $r_N = \sum_{i=1}^{N} 2^{-|f(i)|}$, where f is a computable enumeration of the domain of U, is computable, increasing and converges to Ω.

▸ CEOmegaTheorem

The CBS drama TV show Numb3rs, http://www.cbs.com/primetime/numb3rs/, season 5; episode 5; scene 6) includes the following dialogue:

> LARRY: *Ah, Charles, my ambulatory reference book. Chaitin's Omega Constant...?*
>
> CHARLIE: *Omega equals .00787499699. Why, what're you working on?* (sees the file, reacts) *Oh. FBI file.*

The math is explained at http://numb3rs.wolfram.com/505.

▸ Incomputability of Omega

References

[1] A. A. Abbott, C. S. Calude, J. Conder, and K. Svozil.
Strong Kochen-Specker theorem and incomputability of quantum randomness.
Physical Review A, 86(062109), Dec 2012.

[2] A. A. Abbott, C. S. Calude, M. J. Dinneen, and N. Huang.
Experimentally probing the algorithmic randomness and incomputability of quantum randomness.
Physica Scripta, 94(4):045103, Feb 2019.

[3] A. A. Abbott, C. S. Calude, and K. Svozil.
Value indefiniteness is almost everywhere.
Physical Review A, 89(3):032109–032116, 2014.

References (cont.)

[4] A. A. Abbott, C. S. Calude, and K. Svozil.
A non-probabilistic model of relativised predictability in physics.
Information, 6(4):773–789, 2015.

[5] M. Agrawal, N. Kayal, and N. Saxena.
Primes is in P.
Annals of Mathematics, 160:781–793, 2004.

[6] J. M. Agüero Trejo and C. S. Calude.
A new quantum random number generator certified by value indefiniteness.
Theoretical Computer Science, 862:3–13, Mar. 2021.

References (cont.)

[7] P. Bierhorst, E. Knill, S. Glancy1, Y. Zhang, A. Mink, S. Jordan, A. Rommal, Y.-K. Liu, B. Christensen, S. W. Nam, M. J. Stevens, and L. K. Shalm.
Experimentally generated randomness certified by the impossibility of superluminal signals.
Nature, 556:223–226, April 2018.

[8] C. Calude.
Information and Randomness—An Algorithmic Perspective.
Springer, Berlin, 2002 (2nd ed.).

[9] C. S. Calude.
Information and Randomness: An Algorithmic Perspective.
Springer-Verlag, Berlin, second edition, 2002.

References (cont.)

[10] C. S. Calude, M. J. Dinneen, M. Dumitrescu, and K. Svozil.
Experimental evidence of quantum randomness incomputability.
Physical Review A, 82, 022102:1–8, 2010.

[11] C. S. Calude, M. J. Dinneen, and C. Shu.
Computing a glimpse of randomness.
Experimental Mathematics, 11(3):361–370, 2002.

[12] C. S. Calude and N. J. Hay.
Every computably enumerable random real is provably computably enumerable random.
Log. J. IGPL, 17(4):351–374, 2009.

[13] W. J. Conover.
Practical Nonparametric Statistics.
John Wiley & Sons, New York, 1999.

References (cont.)

[14] R. Downey and D. Hirschfeldt.
Algorithmic Randomness and Complexity.
Springer, Berlin, 2010.

[15] A. Einstein, B. Podolsky, and N. Rosen.
Can quantum-mechanical description of physical reality be considered complete?
Physical Review, 47(10):777–780, May 1935.

[16] M. Kac.
What is random?
American Scientist, 71:405–406, 1983.

References (cont.)

[17] A. Kulikov, M. Jerger, A. Potočnik, A. Wallraff, and A. Fedorov.
Realization of a quantum random generator certified with the Kochen-Specker theorem.
Phys. Rev. Lett., 119:240501, Dec 2017.

[18] A. K. Lenstra, J. P. Hughes, M. Augier, J. W. Bos, T. Kleinjung, and C. Wachter.
Ron was wrong, Whit is right.
Santa Barbara: IACR: 17, `https://eprint.iacr.org/2012/064.pdf`, 2012.

[19] S. Pironio.
The certainty of randomness.
Nature, 556:176–177, 2018.

References (cont.)

[20] R. Solovay and V. Strassen.
Erratum: A fast Monte-Carlo test for primality.
SIAM Journal on Computing, 7(1):118, 1977.

[21] R. Solovay and V. Strassen.
A fast Monte-Carlo test for primality.
SIAM Journal on Computing, 6(1):84–85, 1977.

[22] R. Solovay and V. Strassen.
A fast Monte-Carlo test for primality.
SIAM Journal on Computing, 6(1):84–85, 1977.
Corrigendum in Ref. [20].

[23] R. Solovay and V. Strassen.
Erratum: A fast Monte-Carlo test for primality.
SIAM Journal on Computing, 7(1):118, 1978.

References (cont.)

[24] A. A. Tokuç.
Fermat Probabilistic Test, May 2021.