RaNDomNESS!

Verónica Becher

Universidad de Buenos Aires & CONICET

28th European Summer School in Logic, Language and Information

Bolzano-Bozen, 23 August, 2016

Azar - aléatoire - Zufall - rasgelelik - satunnaisuuden - slumpmässighet - randomness - aleatorietà

Everyone has an intuitive idea about what is randomness, often associated with "gambling" or "luck".

Azar - aléatoire - Zufall - rasgelelik - satunnaisuuden - slumpmässighet - randomness - aleatorietà

Everyone has an intuitive idea about what is randomness, often associated with “gambling” or “luck”.

Today:

- Is there a mathematical definition of randomness?

Azar - aléatoire - Zufall - rasgelelik - satunnaisuuden - slumpmässighet - randomness - aleatorietà

Everyone has an intuitive idea about what is randomness, often associated with "gambling" or "luck".

Today:

- Is there a mathematical definition of randomness?
- Are there degrees of randomness?

Azar - aléatoire - Zufall - rasgelelik - satunnaisuuden - slumpmässighet - randomness - aleatorietà

Everyone has an intuitive idea about what is randomness, often associated with "gambling" or "luck".

Today:

- Is there a mathematical definition of randomness?
- Are there degrees of randomness?
- Examples of randomness?

Azar - aléatoire - Zufall - rasgelelik - satunnaisuuden - slumpmässighet - randomness - aleatorietà

Everyone has an intuitive idea about what is randomness, often associated with "gambling" or "luck".

Today:

- Is there a mathematical definition of randomness?
- Are there degrees of randomness?
- Examples of randomness?
- Can a computer produce a sequence that is truly random?

Azar - aléatoire - Zufall - rasgelelik - satunnaisuuden - slumpmässighet - randomness - aleatorietà

Everyone has an intuitive idea about what is randomness, often associated with “gambling” or “luck”.

Today:

- Is there a mathematical definition of randomness?
- Are there degrees of randomness?
- Examples of randomness?
- Can a computer produce a sequence that is truly random?
- Randomness ♥ Logic, Language and Information

Lady luck is fickle

Think of 0s and 1s.

A sequence is random if it can not be distinguished from independent tosses of a fair coin.

Lady luck is fickle

Would you believe that these have bee obtained by independent toosses?

111111111111111111111111111111111111111111...

Lady luck is fickle

Would you believe that these have bee obtained by independent toosses?

111111111111111111111111111111111111111111... ✗

Lady luck is fickle

Would you believe that these have bee obtained by independent toosses?

$111111111111111111111111111111111111111111\ldots$ ✗

$0100100010000100000100000010000000100000001\ldots$

Lady luck is fickle

Would you believe that these have bee obtained by independent toosses?

11111111111111111111111111111111111111111111... ✗

01001000100001000001000000100000001000000001... ✗

Lady luck is fickle

Would you believe that these have bee obtained by independent toosses?

111111111111111111111111111111111111111111... ✗

01001000100001000001000000100000001000000001... ✗

100101010110001101110100010010101111001001..

Lady luck is fickle

Would you believe that these have bee obtained by independent toosses?

111111111111111111111111111111111111111111... ✗

01001000100001000001000000100000001000000001... ✗

100101010110001101110100010010101111001001.. ✓

Heads and tails must occur with the same frequency.
Likewise for any combination of heads and tails.

Lady luck is fickle

Would you believe that these have bee obtained by independent toosses?

111111111111111111111111111111111111111111... ✗

01001000100001000001000000100000001000000001... ✗

100101010110001101110100010010101111001001.. ✓

Heads and tails must occur with the same frequency.
Likewise for any combination of heads and tails.
¡Otherwise we would be able to guess it infinitely many times!

Randomness is impossibility to guess, to predict, to abbreviate....

Randomness is impossibility to guess, to predict, to abbreviate....

By whom?

By a human being?

Randomness is impossibility to guess, to predict, to abbreviate....

By whom?

By a human being? Ugh! we can not formalize it.

Randomness is impossibility to guess, to predict, to abbreviate....

By whom?

By a human being? Ugh! we can not formalize it.

By a universal Turing machine ?

Randomness is impossibility to guess, to predict, to abbreviate....

By whom?

By a human being? Ugh! we can not formalize it.

By a universal Turing machine ? Yes, it allows formalization and it yields the purest notion of randomness.

Randomness is impossibility to guess, to predict, to abbreviate....

By whom?

By a human being? Ugh! we can not formalize it.

By a universal Turing machine ? Yes, it allows formalization and it yields the purest notion of randomness.

By finite automata? Yes, it allows formalization and it yields the most basic notion of randomness: normality.

Randomness is impossibility to guess, to predict, to abbreviate....

By whom?

By a human being? Ugh! we can not formalize it.

By a universal Turing machine ? Yes, it allows formalization and it yields the purest notion of randomness.

By finite automata? Yes, it allows formalization and it yields the most basic notion of randomness: normality.

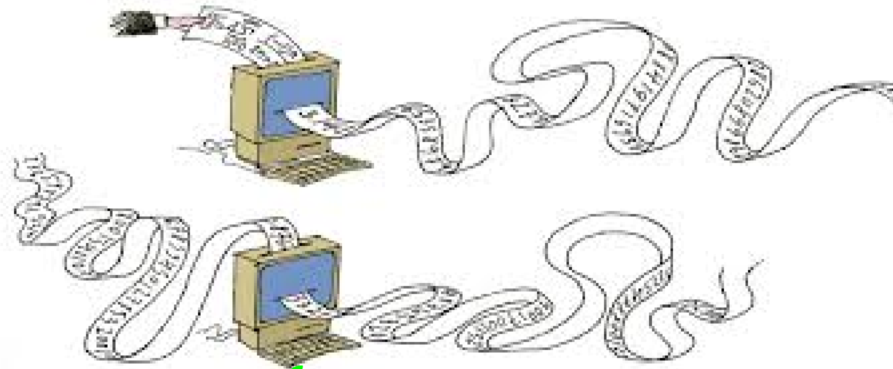
And there are intermediate notions.

Towards a definition of randomness

A sequence is normal if, essentially, its initial segments can only be described explicitely by a finite automaton .

(Borel's definition 1909; Schnorr and Stimm 1971; Dai Lathroup Lutz and Mayordomo 2005)

A sequence is random if, essentially, its initial segments can only be described explicitely by a Turing machine. (Chaitin's definition 1975)



Real numbers and sequences

A base is an integer greater than or equal to 2.

For a real number x in the unit interval, the expansion of x in base b is a sequence $a_1 a_2 a_3 \ldots$ of integers from $\{0, 1, \ldots, b-1\}$ such that

$$x = 0.a_1 a_2 a_3 \ldots$$

where $x = \sum_{k \geq 1} \frac{a_k}{b^k}$, and x does not end with a tail of $b - 1$.

Normal numbers, the most basic form of randomness

Definition (Borel, 1909)

A real number x is simply normal to base b if, in the expansion of x in base b, each digit occurs with limiting frequency equal to $1/b$.

Normal numbers, the most basic form of randomness

Definition (Borel, 1909)

A real number x is simply normal to base b if, in the expansion of x in base b, each digit occurs with limiting frequency equal to $1/b$.

A real number x is normal to base b if, for every positive integer k, every block of k digits (starting at any position) occurs in the expansion of x in base b with limiting frequency $1/b^k$.

Normal numbers, the most basic form of randomness

Definition (Borel, 1909)

A real number x is simply normal to base b if, in the expansion of x in base b, each digit occurs with limiting frequency equal to $1/b$.

A real number x is normal to base b if, for every positive integer k, every block of k digits (starting at any position) occurs in the expansion of x in base b with limiting frequency $1/b^k$.

A real number x is absolutely normal if x is normal to every base.

Not normal

$0.01\ 002\ 0003\ 00004\ 000005\ 0000006\ 00000007\ 000000008\ldots$
is not simply normal to base 10.

Not normal

$0.01\ 002\ 0003\ 00004\ 000005\ 0000006\ 00000007\ 000000008\ldots$
is not simply normal to base 10.

$0.0123456789\ \ 0123456789\ \ 0123456789\ \ 0123456789\ \ 0123456789\ldots$
is simply normal to base 10, but not simply normal to base 100.

Not normal

$0.01\ 002\ 0003\ 00004\ 000005\ 0000006\ 00000007\ 000000008\ldots$
is not simply normal to base 10.

$0.0123456789\ \ 0123456789\ \ 0123456789\ \ 0123456789\ \ 0123456789\ldots$
is simply normal to base 10, but not simply normal to base 100.

The numbers is the middle third Cantor set are not simply normal to base 3 (their expansions lack the digit 1).

Not normal

$0.01\ 002\ 0003\ 00004\ 000005\ 0000006\ 00000007\ 000000008\ldots$
is not simply normal to base 10.

$0.0123456789\ \ 0123456789\ \ 0123456789\ \ 0123456789\ \ 0123456789\ldots$
is simply normal to base 10, but not simply normal to base 100.

The numbers is the middle third Cantor set are not simply normal to base 3 (their expansions lack the digit 1).

The rational numbers are not normal to any base.

Not normal

$0.01\ 002\ 0003\ 00004\ 000005\ 0000006\ 00000007\ 000000008\ldots$
is not simply normal to base 10.

$0.0123456789\quad 0123456789\quad 0123456789\quad 0123456789\quad 0123456789\ldots$
is simply normal to base 10, but not simply normal to base 100.

The numbers is the middle third Cantor set are not simply normal to base 3 (their expansions lack the digit 1).

The rational numbers are not normal to any base.

Liouville's constant $\sum_{n\geq 1} 10^{-n!}$ is not normal to any base.

Examples of normal numbers?

Theorem (Borel 1909)

Almost all real numbers are absolutely normal.

Problem (Borel 1909)

Give one example of an absolutely normal number.

Examples of normal numbers?

Theorem (Borel 1909)

Almost all real numbers are absolutely normal.

Problem (Borel 1909)

Give one example of an absolutely normal number.

Are the usual mathematical constants, such as π, e, or $\sqrt{2}$, absolutely normal? Or at least simply normal to some base?

Examples of normal numbers?

Theorem (Borel 1909)

Almost all real numbers are absolutely normal.

Problem (Borel 1909)

Give one example of an absolutely normal number.

Are the usual mathematical constants, such as π, e, or $\sqrt{2}$, absolutely normal? Or at least simply normal to some base?

Conjecture (Borel 1950)

Irrational algebraic numbers are absolutely normal.

Normal to a given base

Theorem (Champernowne, 1933)

$0.123456789101112131415161718192021\ldots$ *is normal to base* 10.

It is unknown if it is normal to bases that are not powers of 10.

Besicovitch 1935; Copeland and Erdös 1946; ... Ugalde 2000; Alvarez, Becher, Ferrari and Yuhjtman 2016.

Absolutely normal

Sierpinski 1917, Lebesgue 1917; Turing 1937; Schmidt 1961; M. Levin 1970; ... Lutz and Mayordomo 2013; Figueira and Nies 2013.

Theorem (Becher, Heiber and Slaman, 2013)

There is an algorithm that computes an absolutely normal number with just above quadratic time-complexity.

Absolutely normal

Sierpinski 1917, Lebesgue 1917; Turing 1937; Schmidt 1961; M. Levin 1970; ... Lutz and Mayordomo 2013; Figueira and Nies 2013.

Theorem (Becher, Heiber and Slaman, 2013)

There is an algorithm that computes an absolutely normal number with just above quadratic time-complexity.

$0.4031290542003809132371428380827059102765116777624189775110896366\ldots$

Normal to some bases and not to others

Theorem (Cassels 1959; Schmidt 1961)

Almost all numbers in the Cantor ternary set are normal to base 2.

Normal to some bases and not to others

Theorem (Cassels 1959; Schmidt 1961)

Almost all numbers in the Cantor ternary set are normal to base 2.

Theorem (Bailey and Borwein 2012)

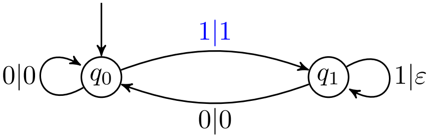
Stoneham number $\alpha_{2,3} = \sum_{k \geq 1} \frac{1}{3^k \; 2^{3^k}}$ *is normal to base* 2 *but not simply normal to base* 6.

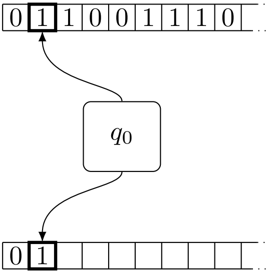
Normality and finite automata

Normality and finite automata

A deterministic finite transducer T is defined by $\langle Q, A, \delta, q_0 \rangle$ where A is the alphabet, Q is a finite set of states with q_0 the starting state, and $\delta : Q \times A \to A^* \times Q$ is a transition function.
Every infinite run is accepting (Büchi acceptance condition).

Running T with input $a_1 a_2 a_3 \ldots$ gives $T(a_1 a_2 a_3 \ldots)$.

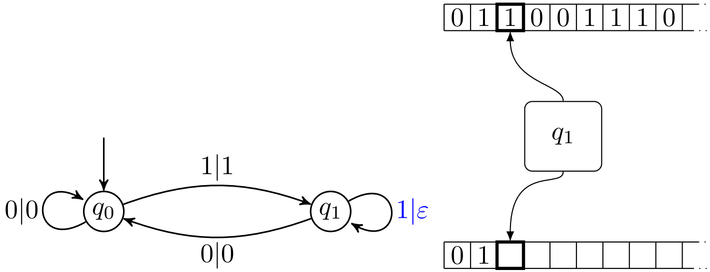




Normality and finite automata

A deterministic finite transducer T is defined by $\langle Q, A, \delta, q_0 \rangle$ where A is the alphabet, Q is a finite set of states with q_0 the starting state, and $\delta : Q \times A \to A^* \times Q$ is a transition function.
Every infinite run is accepting (Büchi acceptance condition).

Running T with input $a_1 a_2 a_3 \ldots$ gives $T(a_1 a_2 a_3 \ldots)$.

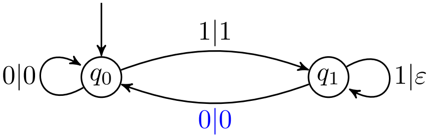


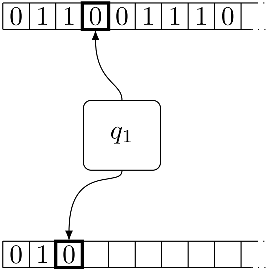
The transducer transforms rows of 1s into a single 1.

Normality and finite automata

A deterministic finite transducer T is defined by $\langle Q, A, \delta, q_0 \rangle$ where A is the alphabet, Q is a finite set of states with q_0 the starting state, and $\delta : Q \times A \to A^* \times Q$ is a transition function.
Every infinite run is accepting (Büchi acceptance condition).

Running T with input $a_1 a_2 a_3 \ldots$ gives $T(a_1 a_2 a_3 \ldots)$.



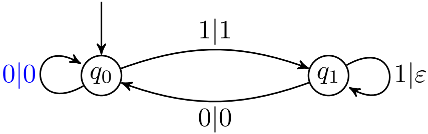


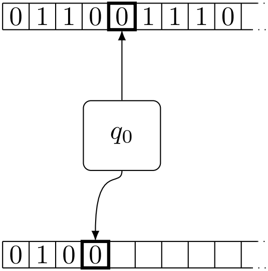
The transducer transforms rows of 1s into a single 1.

Normality and finite automata

A deterministic finite transducer T is defined by $\langle Q, A, \delta, q_0 \rangle$ where A is the alphabet, Q is a finite set of states with q_0 the starting state, and $\delta : Q \times A \to A^* \times Q$ is a transition function.
Every infinite run is accepting (Büchi acceptance condition).

Running T with input $a_1 a_2 a_3 \ldots$ gives $T(a_1 a_2 a_3 \ldots)$.



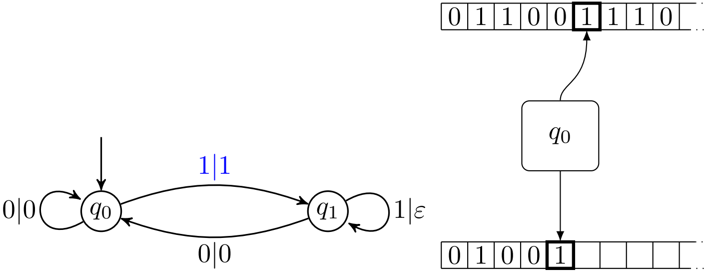


The transducer transforms rows of 1s into a single 1.

Normality and finite automata

A deterministic finite transducer T is defined by $\langle Q, A, \delta, q_0 \rangle$ where A is the alphabet, Q is a finite set of states with q_0 the starting state, and $\delta : Q \times A \to A^* \times Q$ is a transition function.
Every infinite run is accepting (Büchi acceptance condition).

Running T with input $a_1 a_2 a_3 \ldots$ gives $T(a_1 a_2 a_3 \ldots)$.

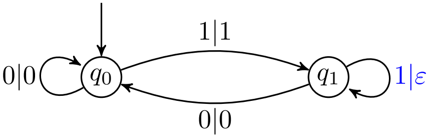


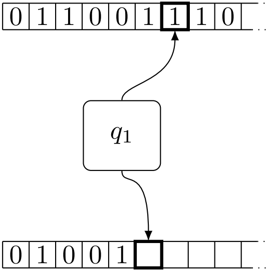
The transducer transforms rows of 1s into a single 1.

Normality and finite automata

A deterministic finite transducer T is defined by $\langle Q, A, \delta, q_0 \rangle$ where A is the alphabet, Q is a finite set of states with q_0 the starting state, and $\delta : Q \times A \to A^* \times Q$ is a transition function.
Every infinite run is accepting (Büchi acceptance condition).

Running T with input $a_1 a_2 a_3 \ldots$ gives $T(a_1 a_2 a_3 \ldots)$.



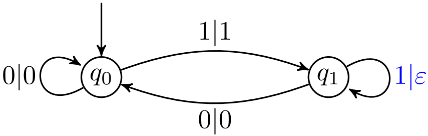


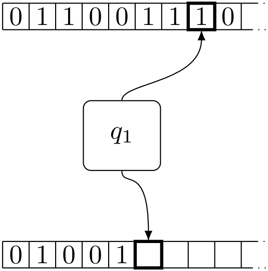
The transducer transforms rows of 1s into a single 1.

Normality and finite automata

A deterministic finite transducer T is defined by $\langle Q, A, \delta, q_0 \rangle$ where A is the alphabet, Q is a finite set of states with q_0 the starting state, and $\delta : Q \times A \to A^* \times Q$ is a transition function.
Every infinite run is accepting (Büchi acceptance condition).

Running T with input $a_1 a_2 a_3 \ldots$ gives $T(a_1 a_2 a_3 \ldots)$.



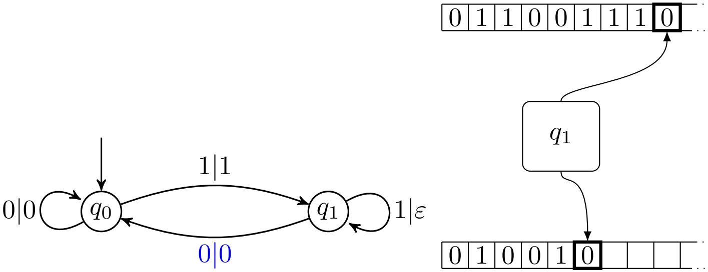


The transducer transforms rows of 1s into a single 1.

Normality and finite automata

A deterministic finite transducer T is defined by $\langle Q, A, \delta, q_0 \rangle$ where A is the alphabet, Q is a finite set of states with q_0 the starting state, and $\delta : Q \times A \to A^* \times Q$ is a transition function.
Every infinite run is accepting (Büchi acceptance condition).

Running T with input $a_1 a_2 a_3 \ldots$ gives $T(a_1 a_2 a_3 \ldots)$.



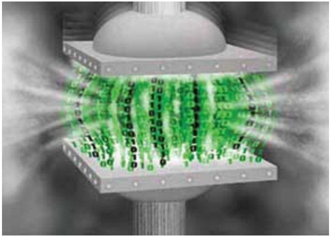
The transducer transforms rows of 1s into a single 1.

Normality and finite automata

Consider transducer $T = \langle Q, A, \delta, q_0 \rangle$. If $\delta(p, a) = \langle v, q \rangle$ write $p \xrightarrow{a|v} q$.

Definition

A sequence $x = a_1 a_2 a_3 \cdots$ is compressible by a finite transducer T if and only if the run in T $q_0 \xrightarrow{a_1|v_1} q_1 \xrightarrow{a_2|v_2} q_2 \xrightarrow{a_3|v_3} q_3 \cdots$ satisfies
$\liminf_{n\to\infty} \frac{|v_1 v_2 \cdots v_n|}{n} < 1.$



Recall that the a's are symbols and the v's are words, possibly empty.

Normality and finite automata

Theorem (Schnorr, Stimm 1971; Dai, Lathrop, Lutz, Mayordomo 2004)

A sequence is normal if and only if it is incompressible by every one-to-one finite transducer .

Huffman 1959 calls them lossless compressors. A direct proof in Becher and Heiber, 2012.

Normality and finite automata

Theorem (Schnorr, Stimm 1971; Dai, Lathrop, Lutz, Mayordomo 2004)

A sequence is normal if and only if it is incompressible by every one-to-one finite transducer .

Huffman 1959 calls them lossless compressors. A direct proof in Becher and Heiber, 2012.

Theorem (Becher, Carton, Heiber 2013)

Non-deterministic one-to-one finite transducers, even if augmented with a counter, can not compress normal sequences.

Normality and pushdown automata

Question

Can deterministic pushdown transducers compress normal infinite sequences?

Normality and pushdown automata

Question

Can deterministic pushdown transducers compress normal infinite sequences?

Theorem (Boasson, personal communication 2012)

Non-deterministic puhdown transducers can compress normal sequences.

0123456789 9876543210 00 01 02 03 ...98 99 99 98 97...03 02 01 00 000 001 002...

Pure randomness

Pure randomness

A sequence is random if its initial segments can only be described explicitely by a Turing machine. That is, its initial segments cannot be compressed with a Turing machine.

Pure randomness

A sequence is random if its initial segments can only be described explicitely by a Turing machine. That is, its initial segments cannot be compressed with a Turing machine.

Formally, a sequence is random if its initial segments have almost maximal program-size complexity .

Kolmogorov / program-size complexity

Some long strings can be described using fewer symbols than their length; this is used in data compression .

For example, string consisting of 2^n many a's can be encoded as $\log n$ many symbols plus a constant:

```
input n
i=0;
while (i<2^n) {print a; i=i+1;}
```

Kolmogorov / program-size complexity

Definition (Kolmogorov 1965)

Fix a universal Turing machine U. The Kolmogorov complexity of a string s is the length of the shortest input in U that outputs s.

Kolmogorov / program-size complexity

Definition (Kolmogorov 1965)

Fix a universal Turing machine U. The Kolmogorov complexity of a string s is the length of the shortest input in U that outputs s.

For every string s, its Kolmogorov complexity is less than $|s| + constant$.

Kolmogorov / program-size complexity

Definition (Kolmogorov 1965)

Fix a universal Turing machine U. The Kolmogorov complexity of a string s is the length of the shortest input in U that outputs s.

For every string s, its Kolmogorov complexity is less than $|s| + constant$.

Definition (Chaitin 1975)

Fix a universal Turing machine U with prefix-free domain .
The program-size complexity of a string s, $K(s)$, is the length of the shortest input in U that outputs s.

For every string s, $K(s) \leq |s| + 2 \log |s| + constant$.

The definition of randomness

Definition (Chaitin 1975)

A sequence $a_1a_2a_3\ldots$ is random if $\exists c\ \forall n\ K(a_1a_2\ldots a_n) > n - c$.

The definition applies immediately to real numbers (one-to-one correspondence between reals and their expansions in any given base).

How do we know that the definition is right?

How do we know that the definition is right?

The definition of randomness was accepted when two different formulations were shown to be equivalent.

This is similar to what happenned with the notion of algorithm in 1930s with Church-Turing thesis.

An equivalent definition of randomness

Definition (Martin-Löf 1965, tests of non-randomness)

A sequence is Martin-Löf random if it passes all computably definable tests of non-randomness. Since there is a universal tests, it suffices that to consider just this universal Martin-Löf test.

An equivalent definition of randomness

Definition (Martin-Löf 1965, tests of non-randomness)

A sequence is Martin-Löf random if it passes all computably definable tests of non-randomness. Since there is a universal tests, it suffices that to consider just this universal Martin-Löf test.

Technically, a sequence is Martin-Löf random if it belongs to no computably definable null set. Since there is a universal computably definable null set, it suffices to consider this one.

An equivalent definition of randomness

Definition (Martin-Löf 1965, tests of non-randomness)

A sequence is Martin-Löf random if it passes all computably definable tests of non-randomness. Since there is a universal tests, it suffices that to consider just this universal Martin-Löf test.

Technically, a sequence is Martin-Löf random if it belongs to no computably definable null set. Since there is a universal computably definable null set, it suffices to consider this one.

Theorem (Schnorr 1975)

A sequence is random for Chaitin's definition if and only if it does not belong to the universal Martin-Löf null set.

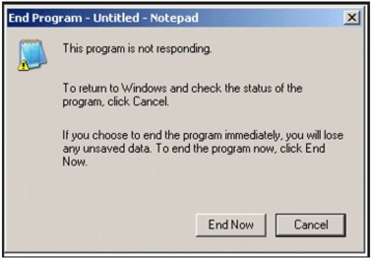
Examples of random sequences

Examples of random sequences

Have you ever experienced that your computer locked up (froze)?

Examples of random sequences

Have you ever experienced that your computer locked up (froze)?



Ω-numbers

Theorem (Chaitin 1975)

The probability that a universal Turing machine with prefix-free domain halts,
$$\Omega = \sum_{U(p)\, \textit{halts}} 2^{-|p|}$$ *is random.*

Simliarly, probabilities of other computer behaviours called Ω numbers (Becher,Chaitin 2001,2003; Becher,Grigorieff 2005,2009, Becher,Figueira,Grigorieff,Miller 2006; Barmpalias 2016)

Questions and answers about random sequences

Are almost all sequences random?

Questions and answers about random sequences

Are almost all sequences random?

Yes. By definition, the set of random sequences is the whole set minus the effectively defined universal null set. Then, with probability 1 an arbitrary sequence belongs to the set of random sequences.

Questions and answers about random sequences

Are random sequences normal?

Questions and answers about random sequences

Are random sequences normal?

Yes. Incompressibility by a Turing machine imples incompressibility by a finite automaton.

Questions and answers about random sequences

Are random sequences normal?

Yes. Incompressibility by a Turing machine imples incompressibility by a finite automaton.

Yes. Another proof: The set of non-normal sequences is properly included in a computably definable null set.

Questions and answers about random sequences

Is the spell of good luck (or bad luck) necessarily short?

Questions and answers about random sequences

Is the spell of good luck (or bad luck) necessarily short?

Yes ("Nothing lasts forever. . . ").

Proof: Think of 0s and 1s. Suppose a random sequence starts $a_1a_2...a_n$. If there is a run of 0's longer than $\log n$, then $a_1a_2...a_n$ is compressible. Randomness ensures that this will happen only finitely many times.

Questions and answers about random sequences

Can a computer output a random sequence?

Questions and answers about random sequences

Can a computer output a random sequence?

"Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin."

John Von Neumann (1951). Various techniques used in connection with random digits.

Questions and answers about random sequences

Can a computer output a random sequence?

"Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin."

John Von Neumann (1951). Various techniques used in connection with random digits.

Proof: Every computable sequences is dramatically compressible by a Turing machine! An initial segment of length n can be compressed to $2\log n$+constant. Hence, computable sequences are not random.

Randomness ☠ Computers

Random number generators (pseudo randomness)
USA National Institute of Standards and Technology
`http://csrc.nist.gov/groups/ST/toolkit/rng/`

Randomness ☠ Computers

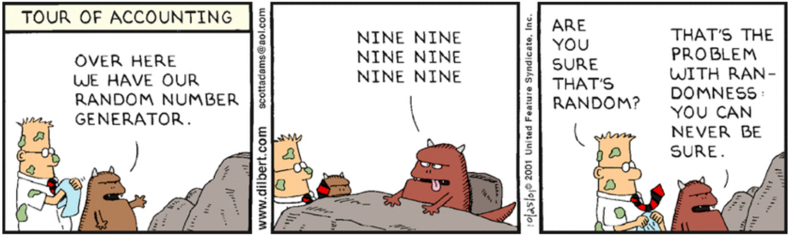
Random number generators (pseudo randomness)
USA National Institute of Standards and Technology
`http://csrc.nist.gov/groups/ST/toolkit/rng/`

`http://www.random.org/`

Randomness ☠ Computers

Random number generators (pseudo randomness)
USA National Institute of Standards and Technology
`http://csrc.nist.gov/groups/ST/toolkit/rng/`

`http://www.random.org/`



Randomness ♥ Logic

Randomness ♥ Logic

The Berry's paradox

Randomness ♥ Logic

The Berry's paradox

Give the smallest positive integer not definable in fewer than thirteen words.

Randomness ♥ Logic

The Berry's paradox

Give the smallest positive integer not definable in fewer than thirteen words.
The above sentence has twelve.

Randomness ♥ Logic

The Berry's paradox

Give the smallest positive integer not definable in fewer than thirteen words.
The above sentence has twelve.

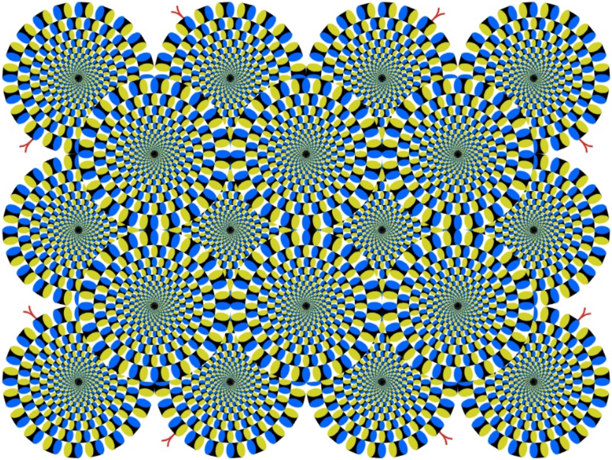
G.G.Berry 1867–1928, librarian at Oxford's Bodleian library.

G.Boolos (1989) built on a formalized version of Berry's paradox to prove Gödel's Incompleteness Theorem formalizing the expression "m is the first number not definable in less than k symbols".

X.Caicedo (1993), La paradoja de Berry revisitada, o la indefinibilidad de la definibilidad y las limitaciones de los formalismos Lecturas Matemáticas 14: 37-48.

Berry's paradox

Though the formal analogue does not lead to a logical contradiction, it yields a proof that Kolmogorov complexity K is not computable.



"Rotating snakes"

Circular snakes appear to rotate 'spontaneously'.

A. Kitaoka, 2003

Randomness ♥ Logic

Theorem

Let U be a universal Turing machine. The function $K(s) = \min\{|t| : U(t) = s\}$ is not computable.

Proof. Assume K is computable. Consider the following program:

```
int main(){
    int K(String s){ ....}
```

Randomness ♥ Logic

Theorem

Let U be a universal Turing machine. The function $K(s) = \min\{|t| : U(t) = s\}$ is not computable.

Proof. Assume K is computable. Consider the following program:

```
int main(){
    int K(String s){ ....}
    const C = 10000; /* greater than or equal to this program length*/
```

Randomness ♥ Logic

Theorem

Let U be a universal Turing machine. The function $K(s) = \min\{|t| : U(t) = s\}$ is not computable.

Proof. Assume K is computable. Consider the following program:

```
int main(){
    int K(String s){ ....}
    const C = 10000; /* greater than or equal to this program length*/
    String s=empty word;
    while (K(s) ≤ C) s= next(s);
    print s;
```

Randomness ♥ Logic

Theorem

Let U be a universal Turing machine. The function $K(s) = \min\{|t| : U(t) = s\}$ is not computable.

Proof. Assume K is computable. Consider the following program:

```
int main(){
     int K(String s){ ....}
     const C = 10000; /* greater than or equal to this program length*/
     String s=empty word;
     while (K(s) ≤ C) s= next(s);
     print s;
}
```

According to the execution $K(output) > C$.

However,

Randomness ♥ Logic

Theorem

Let U be a universal Turing machine. The function $K(s) = \min\{|t| : U(t) = s\}$ *is not computable.*

Proof. Assume K is computable. Consider the following program:

```
int main(){
    int K(String s){ ....}
    const C = 10000; /* greater than or equal to this program length*/
    String s=empty word;
    while (K(s) ≤ C) s= next(s);
    print s;
}
```

According to the execution $K(output) > C$.

However, $K(output) \leq |\texttt{int main()}\{...\}| \leq C$.

Randomness ♥ Logic

Theorem

Let U be a universal Turing machine. The function $K(s) = \min\{|t| : U(t) = s\}$ is not computable.

Proof. Assume K is computable. Consider the following program:

```
int main(){
    int K(String s){ ....}
    const C = 10000; /* greater than or equal to this program length*/
    String s=empty word;
    while (K(s) ≤ C) s= next(s);
    print s;
}
```

According to the execution $K(output) > C$.

However, $K(output) \leq |\texttt{int main()}\{...\}| \leq C$.

Contradiction.

Randomness ♥ Information

Randomness ♥ Information

Program-size complexity is formally identical to Shannon's Information Theory

Randomness ♥ Information

Definition (Shannon 1948)

Given a probability P of a discrete random variable X, the entropy $H(X) = \sum_x P(x = X)(-\log P(x = X))$.

Definition (Chaitin 1975)

Fix a universal Turing U machine with prefix-free domain.
$K(s) = \min\{|t| : U(t) = s\}$, $P(s) = \sum_{t:U(t)=s} 2^{-|t|}$.

Theorem (Chaitin 1975)

For every string s, $K(s) \simeq \lceil -\log P(s) \rceil$.

Thus, entropy is essentially expected program-size complexity :

$$\sum_s P(s)(-\log P(s)) \simeq \sum_s P(s)K(s).$$

Randomness ♥ Language

Randomness ♥ Language

A sequence is random (relative to some computing power) if, essentially, the only way to describe it is explicitely.

Randomness ♥ Language

A sequence is random (relative to some computing power) if, essentially, the only way to describe it is explicitely.

Therefore, randomness of a given sequence is about how we can describe its initial segments in the language , according to the computing power.

Randomness ♥ Language

A sequence is random (relative to some computing power) if, essentially, the only way to describe it is explicitely.

Therefore, randomness of a given sequence is about how we can describe its initial segments in the language , according to the computing power.

Thus, randomness is a matter of language.

Randomness ♥ Language

A sequence is random (relative to some computing power) if, essentially, the only way to describe it is explicitely.

Therefore, randomness of a given sequence is about how we can describe its initial segments in the language , according to the computing power.

Thus, randomness is a matter of language.

The End

RaNDomNEss!