Exploring randomness computably

A short introduction to Kolmogorov complexity

Martin Trucchi j.w.w. Dominik Kirst & Yannick Forster

soon-to-be Inria Paris

19 septembre 2025

Presented at PC25

A small example

Which of these sequences look the most random ?

A small example

Which of these sequences look the most random ?

- 00000000000000000000
-
-
-

A small example

Which of these sequences look the most random ?

- 00000000000000000000
- 11010011111001001101
-
-

A small example

Which of these sequences look the most random ?

- 00000000000000000000
- 11010011111001001101
- 01010101010101010101
-

A small example

Which of these sequences look the most random ?

- 00000000000000000000
- 11010011111001001101
- 01010101010101010101
- 10100100010000100000

A small example

Which of these sequences look the most random ?

- ~~00000000000000000000~~
- 11010011111001001101
- 01010101010101010101
- 10100100010000100000

A small example

Which of these sequences look the most random ?

- ~~00000000000000000000~~
- 11010011111001001101
- ~~01010101010101010101~~
- 10100100010000100000

A small example

Which of these sequences look the most random ?

- ~~00000000000000000000~~
- 11010011111001001101
- ~~01010101010101010101~~
- ~~10100100010000100000~~

A small example

Which of these sequences look the most random ?

- ~~00000000000000000000~~
- 11010011111001001101
- ~~01010101010101010101~~
- ~~10100100010000100000~~

Kolmogorov complexity

We consider chains (= lists) of booleans.

Kolmogorov complexity

We consider chains (= lists) of booleans.

Kolmogorov complexity is a measure of the quantity of "raw" information contained in a chain.

Kolmogorov complexity

We consider chains (= lists) of booleans.

Kolmogorov complexity is a measure of the quantity of "raw" information contained in a chain.

Intuitively the size of the smallest program computing the chain.

Back to examples

00000000000000000000

```
def chain(n):
    return 0
```

Back to examples

01010101010101010101

```
def chain(n):
    if (n%2 == 0):
        return 0
    else:
        return 1
```

Back to examples

10100100010000100000

```
def chain(n):
    c = 0 ; k = 0
    while (c < n):
        k += 1
        c += k + 1
    return (int(c == n))
```

Back to examples

```
11010011111001001101
```

```
def chain(n):
    return("11010011111001001101"[n])
```

Formal definition

Definition: Kolmogorov complexity [3]

Let U be a fixed universal prefix-free Turing machine. We define the Kolmogorov complexity of a chain $\sigma \in 2^*$ by :
$K(\sigma) = \min\{|\tau| \mid U(\tau) \downarrow = \sigma\}$

Randomness of a binary stream

Definition: Randomness [3]

Let $s \in 2^\omega$ be a binary stream. We say that s is random if :
$\exists c \in \mathbb{N}, \forall n \in \mathbb{N}, K(s_{\restriction n}) > n - c$

Formalization & computability

- This notion is strongly connected to computability theory.

Formalization & computability

- This notion is strongly connected to computability theory.
- We aim to formalize results about Kolmogorov complexity in Rocq, continuing work from [2] and [1]

Formalization & computability

- This notion is strongly connected to computability theory.
- We aim to formalize results about Kolmogorov complexity in Rocq, continuing work from [2] and [1]
- The framework is abstract, and avoid talking about technical Turing machines.

Conclusion

Thank you for listening !

References I


Yannick Forster.
Parametric church's thesis : Synthetic computability without choice.
In Sergei Artemov and Anil Nerode, editors, Logical Foundations of Computer Science, pages 70–89, Cham, 2022. Springer International Publishing.


Yannick Forster, Fabian Kunze, and Nils Lauermann.
Synthetic Kolmogorov Complexity in Coq.
In 13th International Conference on Interactive Theorem Proving (ITP 2022), volume 237, pages 12 :1–12 :19, 2022.


Benoît Monin and Ludovic Patey.
Complexité de kolmogorov et nombres aléatoires.
In Calculabilité, chapter 16. Calvage Mounet, 2022.