

# Kolmogorov complexity, randomness and why not everything is possible

---

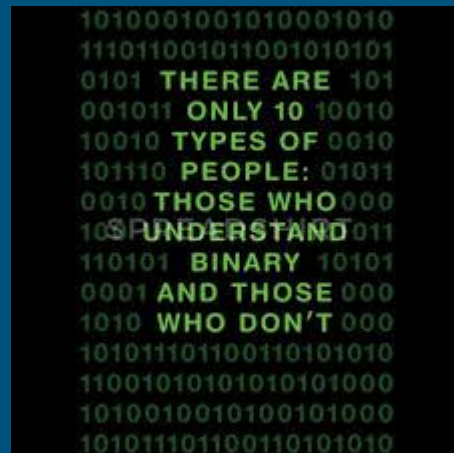
with applications in cryptography and  
data compression

# Preliminary notes

---

For the purpose of this presentation, we will be talking about binary strings only. Simply because we can convert pretty much any piece of information into binary strings.

For non Java people, Java is used throughout this presentation.



```
1010001001010001010
111011001011001010101
0101 THERE ARE 101
001011 ONLY 10 10010
10010 TYPES OF 0010
101110 PEOPLE: 01011
0010 THOSE WHO 000
10101101101101101101
110101 BINARY 10101
0001 AND THOSE 000
1010 WHO DON'T 000
101011101100110101010
11001010101010101000
1010010010100101000
101011101100110101010
```

```
1010101010101010101010101010101010101010101010101010101010101010  
00011101001000101101001000101111010100000100111101
```



# What is randomness?

10101010101010101010101010101010101010101010101010101010  
00011101001000101101001000101111010100000100111101

No, all it can say is, they both have the same chance of

$$P = \frac{1}{2^{50}}$$

But we can definitely see some structure with the first string and less with the second.



# Compression to measure randomness

---

And the structure is

```
range(0, 25).foreach(e -> out.print("10"));
```

I used 43 bytes to produce 50 (Yes, I cheated!).

Randomness intuition (aka Kolmogorov Randomness): A string of bits is random if and only if it is shorter than any computer program that can reproduce that string.

This means that random strings are those that cannot be compressed.

# Kolmogorov complexity

---

DEF: Kolmogorov complexity of a string  $x$ , relative to a Turing machine  $f$ , is defined as

$$K_f(x) = \min\{|p| \mid f(p) = x\}$$

$|p|$  is the length of the program (yes, as a binary string) in Turing machine  $f$ .

Why Turing machines?

Well, all computational models are equivalent.



# Kolmogorov complexity

The choice of Turing machine doesn't matter, the result will differ by a constant  $C$

$$\forall x, \quad |K_{f_1}(x) - K_{f_2}(x)| \leq C$$

And of course

$$\forall x, \quad K_f(x) \leq |x| + C \quad \text{or} \quad K_f(x) \leq \log_2 x + C$$

because we can always `System.out.print(x)` and for example

$$|(1111)_2| = \lceil \log_2 (15)_{10} \rceil = \lceil 3.906.. \rceil = 4$$

# Kolmogorov complexity

---

There is a definition for infinite strings  $x$ , basically dealing with the first  $n$  bits of the string or prefix. But we will skip this part.





# And the randomness is ...

---

DEF: The string  $x$  is incompressible, random, Kolmogorov random, algorithmically random if  $K(x) \geq |x|$

I.e. the shortest way to reproduce the string is to print it.

NOTE: I omitted  $f$ , since we can always consider a Turing machine producing minimal code instead.

DEF: The string  $x$  is badly compressible or  $c$ -compressible ( $c$  - some constant) if

$$K(x) \geq |x| - c$$

# Randomness exists and it's huge

It is obvious if we look at the cardinality of all strings of length  $n$  vs cardinality of strings of length  $< n$ . There is no one to one mapping.

$$1 + 2 + 2^2 + \dots + 2^{n-1} = 2^n - 1 \quad \text{vs} \quad 2^n$$

As with 1-compressible strings

$$K(x) \geq n - 1 \Rightarrow \\ 1 + 2 + 2^2 + \dots + 2^{n-2} = 2^{n-1} - 1 \quad \text{vs} \quad 2^n - 2^{n-1} + 1 = 2^{n-1} + 1$$

Basically, there is more **randomness** than structure!



# One lesson learned

---

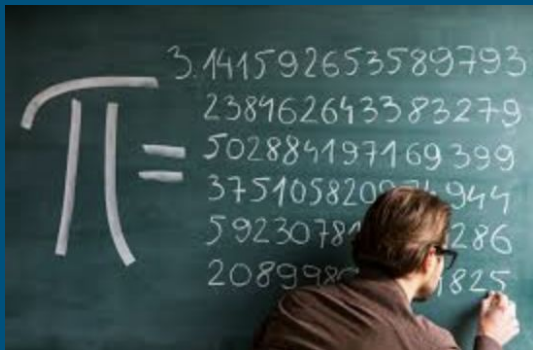
If somebody tries to sell you a program capable to compress everything

**DON'T BUY IT !!!**



# Another lesson

Is  $\pi$  random?



$$\pi = \sum_{k=0}^{\infty} \frac{1}{16^k} \left( \frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6} \right)$$

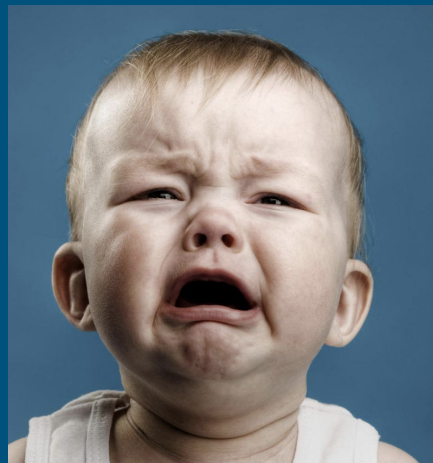
No, we can write a *finite* program which can produce any hexadecimal digit of it.

# Bad news

---

Kolmogorov complexity is **uncomputable**. That is, there is no Turing machine which can compute it for every possible string.

The reason for that is quite simple, if  $K(.)$  was computable then there would be a finite program of size  $n$  capable to compute complexity of strings with complexity  $> n$  (like random strings of length  $n+1$  as we saw previously). Because of this finite program, this would mean those strings have complexity  $\leq n$ . So they aren't random. Contradiction.



# Some exotic examples

---

Berry paradox: what is “the smallest natural number that cannot be described in eleven words”.

The sentence itself contains 11 words. We could define the set of natural numbers which cannot be described in 11 words. Each subset of natural numbers has a minimum. This minimum is “the smallest natural number that cannot be described in eleven words”, but it is described by this sentence, which has 11 words. Computing words to describe natural numbers is **uncomputable**, paradox solved!

# Some exotic examples

Gödel incompleteness theorem in Kolmogorov's complexity language:

There is a number  $m$  such that for every string  $x$  the proposition

$$K(x) \geq m$$

is unprovable.



# Why randomness is important?

Well, there are money prizes, like the million random digit challenge (link in *Additional reading*), offering \$100!

But, on a serious note, randomness is heavily used in **cryptography** and related fields ...





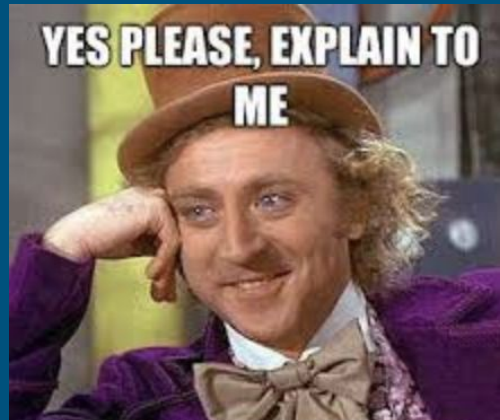
# Randomness in cryptography

---

If a cipher, produced by a cryptographic function, is well compressible, then this function is bad.

Key generators for symmetric ciphers must produce random keys.

HOLD ON! But we cannot verify randomness since Kolmogorov complexity is ***uncomputable!***



# Randomness in cryptography

Indeed, cryptography had to invent what are known as PRG (pseudo random generator), statistical tests and unpredictability.

PRG is a deterministic function

$$G : \{0, 1\}^k \rightarrow \{0, 1\}^n, \quad n \gg k$$

PRG is unpredictable if for any  $i$

$$P(x_{i+1} \mid x_1, x_2, \dots, x_i) = \frac{1}{2}$$

Statistical test is a program (algorithm)

$$A : \{0, 1\}^n \rightarrow \{0, 1\} \text{ such that } A(x) = \begin{cases} 1 & x - \text{random} \\ 0 & \text{otherwise} \end{cases}$$



# Randomness in cryptography

---

PRG are not random in Kolmogorov sense. Since PRG are small programs capable of generating large *statistical-random* strings. These strings have small Kolmogorov complexity (due to the size of the program).

If  $P=NP$  is provable, then there is no secure PRG. If secure PRG exists, then  $P \neq NP$ .



# Summary please?

---

Incompressible strings (randomness) exists.

We can't always decide if a string is random.

The problem of secure PRG is undecided and may never be decided.

So ...

# Next time I hear “Everything is possible!”

---

I reply ...



# Additional reading

---

<https://www.labri.fr/perso/zvonkin/Research/kolmathan.pdf>

<https://brilliant.org/wiki/kolmogorov-complexity/>

<https://jeremykun.com/2012/04/21/kolmogorov-complexity-a-primer/>

<https://marknelson.us/posts/2006/06/20/million-digit-challenge.html>

<http://www.drdoobs.com/architecture-and-design/the-million-random-digit-challenge/228701653>

[https://www.youtube.com/watch?v=fh1vNc4sgml&index=10&list=PL58C6Q25sEEHXvACYxiav\\_IC2DqSIC7Og](https://www.youtube.com/watch?v=fh1vNc4sgml&index=10&list=PL58C6Q25sEEHXvACYxiav_IC2DqSIC7Og)

<https://www.i-programmer.info/babbages-bag/6319-kolmogorov-complexity.html?start=1>