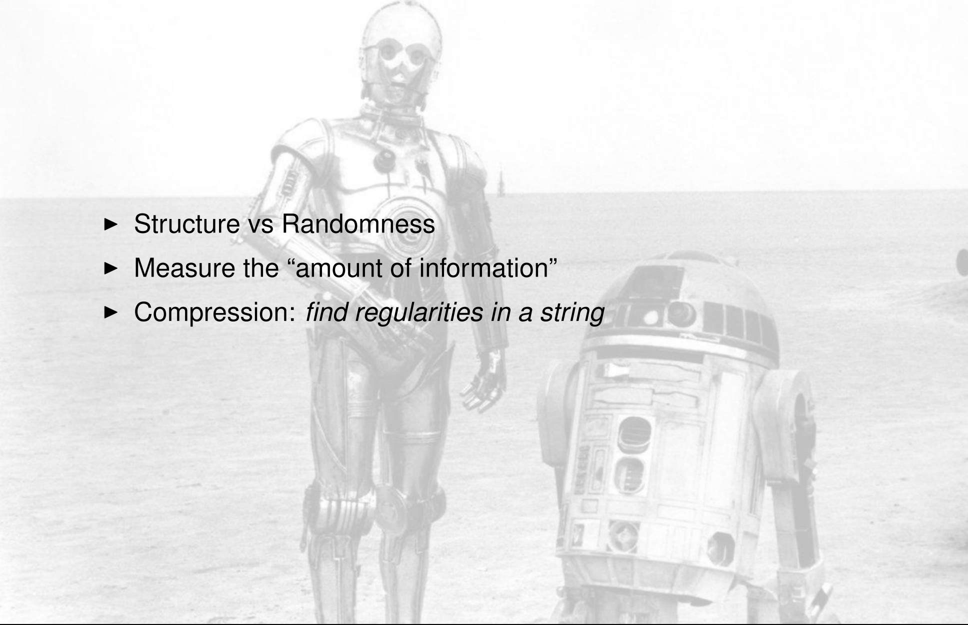
STAR WARS

Episode VIIa : Algorithmic Information Theory

Antonis Antonopoulos

May 19, 2017



- Structure vs Randomness
- Measure the "amount of information"
- Compression: *find regularities in a string*

- How random is $\underbrace{111\cdots1}_{\text{1000 times}}$?

```
for i in range(1, 1000):
    print i
```

- How random is 6535897932384626433832795028841971693?
- $\pi = 3.1415926535897932384626433832795028841971693 20\ldots$

```
def pi(approx):
    result = 0.0
    for n in range(approx):
        result += (-1.0)**n/(2.0*n+1.0)
    return 4*result
```

- How random is $\underbrace{111\cdots1}_{\text{1000 times}}$?

```
for i in range(1, 1000):
    print i
```

- How random is 6535897932384626433832795028841971693?
- $\pi = 3.14159265358979323846264338327950288419716939320\ldots$

```
def pi(approx):
    result = 0.0
    for n in range(approx):
        result += (-1.0)**n/(2.0*n+1.0)
    return 4*result
```

- How random is $\underbrace{111\cdots1}_{1000\text{ times}}$?

```
for i in range(1, 1000):
    print i
```

- How random is 6535897932384626433832795028841971693?
- $\pi = 3.141592653589793238462643383279502884197169320\ldots$

```
def pi(approx):
    result = 0.0
    for n in range(approx):
        result += (-1.0)**n/(2.0*n+1.0)
    return 4*result
```

- How random is $\underbrace{111\cdots1}_{\text{1000 times}}$?

```
for i in range(1, 1000):
    print i
```

- How random is 6535897932384626433832795028841971693?
- $\pi = 3.141592$6535897932384626433832795028841971693$20\ldots$

```
def pi(approx):
    result = 0.0
    for n in range(approx):
        result += (-1.0)**n/(2.0*n+1.0)
    return 4*result
```

- How random is $\underbrace{111\cdots1}_{\text{1000 times}}$?

```
for i in range(1, 1000):
    print i
```

- How random is 6535897932384626433832795028841971693?
- $\pi = 3.141592\textcolor{red}{6535897932384626433832795028841971693}20\ldots$

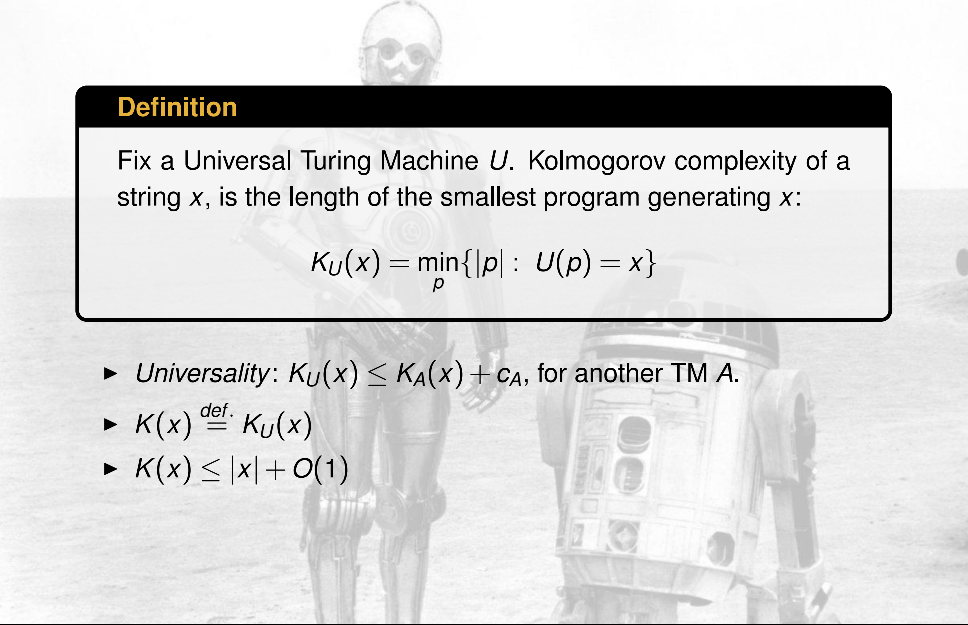
```
def pi(approx):
    result = 0.0
    for n in range(approx):
        result += (-1.0)**n/(2.0*n+1.0)
    return 4*result
```

Definition

Fix a Universal Turing Machine U. Kolmogorov complexity of a string x, is the length of the smallest program generating x:

$$K_U(x) = \min_p \{|p| : U(p) = x\}$$

- Universality: $K_U(x) \leq K_A(x) + c_A$, for another TM A.
- $K(x) \stackrel{def}{=} K_U(x)$
- $K(x) \leq |x| + O(1)$

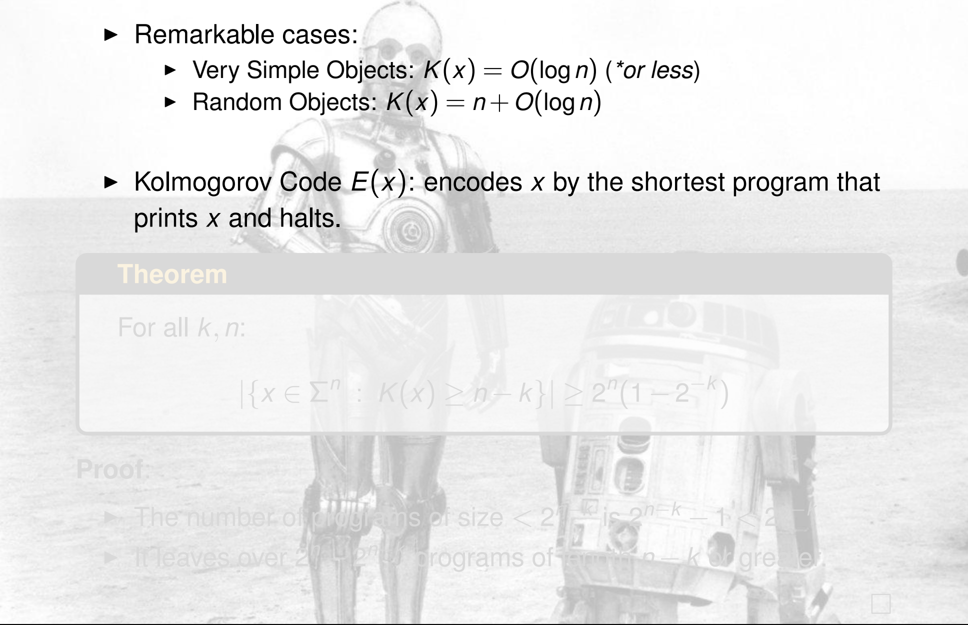


Definition

Fix a Universal Turing Machine U. Kolmogorov complexity of a string x, is the length of the smallest program generating x:

$$K_U(x) = \min_p \{|p| : U(p) = x\}$$

- *Universality*: $K_U(x) \leq K_A(x) + c_A$, for another TM A.
- $K(x) \stackrel{def.}{=} K_U(x)$
- $K(x) \leq |x| + O(1)$



- Remarkable cases:
  - Very Simple Objects: $K(x) = O(\log n)$ (**or less*)
  - Random Objects: $K(x) = n + O(\log n)$

- Kolmogorov Code $E(x)$: encodes x by the shortest program that prints x and halts.

Theorem

For all k, n:

$$|\{x \in \Sigma^n \ : \ K(x) \geq n - k\}| \geq 2^n(1 - 2^{-k})$$

Proof:

- The number of programs of size $< 2^{n-k}$ is $2^{n-k} - 1 < 2^{n-k}$
- It leaves over $2^n - 2^{n-k}$ programs of length $n - k$ or greater.

□

- Remarkable cases:
  - Very Simple Objects: $K(x) = O(\log n)$ (**or less*)
  - Random Objects: $K(x) = n + O(\log n)$

- Kolmogorov Code $E(x)$: encodes x by the shortest program that prints x and halts.

Theorem

For all k, n:

$$|\{x \in \Sigma^n \ : \ K(x) \geq n - k\}| \geq 2^n(1 - 2^{-k})$$

Proof:

- The number of programs of size $< 2^{n-k}$ is $2^{n-k} - 1 < 2^{n-k}$
- It leaves over $2^n - 2^{n-k}$ programs of length $n - k$ or greater.

□

Theorem

For all n, there exists some x with $|x| = n$ such that $K(x) \geq n$.

Proof:

- Suppose, for the sake of contradiction, that for all x: $K(x) < n$
- Thus, $\forall x \exists p_x : \; U(p_x) = x$, and $|p_x| < n$.
- There are $2^n - 1$ programs of length $< n$.
- If all strings of length n had a program shorter than n, there must be a program producing two different strings. Contradiction.

$\square$

- Such a x is called **Kolmogorov Random**.

A Toy Example

Theorem

There are infinitely many primes.

Proof:

- Suppose for the sake of contradiction that they are finite: $p_1, \ldots, p_k$, $k \in \mathbb{N}$
- Let $m \in \mathbb{N}$ be *Kolmogorov random*, having length n.
- $m = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$.
- We can describe m by $< e_1, \cdots, e_k >$, and we claim that this gives a short descrtiption of m
- $e_i \leq \log m \rightarrow |e_i| \leq \log \log m$
- Since $m \leq 2^{n+1}$, $| < e_1, \cdots, e_k > | \leq 2k \log \log m \leq 2k \log(n+1)$
- So $K(m) \leq 2k \log(n+1) + c$, contradicting $K(m) \geq n$.

There is a disturbance in the Force

Theorem

Kolmogorov complexity ($K : \mathbb{N} \to \mathbb{N}$) is undecidable.

Proof:

- Assume, for the sake of contradiction, that K is computable.
- Then, the function $\psi(m) = \min_{x \in \mathbb{N}} \{x : \ K(x) \geq m\}$ is also computable.
- $K(\psi(m)) \geq m$.
- Since ψ is computable, there exists a program of some fixed size c that on input m outputs $\psi(m)$ and halts.
- So, $K(\psi(m)) \leq |m| + c \leq 2\log m + c \ \Rightarrow m \leq 2\log m + c$. Contradiction.

□

Resource-Bounded Kolmogorov Complexity

Definition

$$C^t(x) = \min_p\{|p| : \ U(p) \text{ outputs } x \text{ in } t(|x|) \text{ steps}\}$$

- Notice that here we measure the amount of time as a function of the *output*, not the input.

Definition (Sipser '83)

$$CD^t(x) = \min_p \left\{ |p| : \begin{array}{l} U(p,x) \text{ accepts} \\ U(p,z) \text{ rejects for all } z \neq x \\ U(p,z) \text{ runs in time at most } t(|z|), \forall z \in \Sigma^* \end{array} \right.$$

- Buhrman, Fortow and Laplante (2002) developed a nondeterministic version CND^t.

Definition (Levin '73)

$$Ct(x) = \min_p \{|p| + \log t : \ U(p) = x \text{ in } t \text{ steps}\}$$

Definition (Allender '01)

$$CT(x) = \min_p \{|p| + t : \ U(p,i) = \text{ the } i^{th} \text{ bit of } x \text{in } t \text{ steps}\}$$

- Allender's definition focus on sublinear time, so we need to modify how U produces the string.

Theorem

For all x:

$$CD^t(x) \leq C^t(x) + O(1)$$

Theorem (Fortnow and Kummer '94)

The following are equivalent:

1. USAT is easy (that is **NP** $=$ **RP** and **P** $=$ **UP**).
2. For every polynomial p there exists a polynomial q and a constant c such that for all x, y:

$$C^q(x|y) \leq CD^p(x|y) + c$$

Definition

We define sets of strings with similar Kolmogorov Complexity:

$$C[f(n),t(n)] = \{x \mid C^t(x) \leq f(n)\}$$

- These classes form well-defined hierarchies, with all the nice properties.

Definition

A language L is P-printable if there exists a polynomial time computable function f such that $f(1^n)$ enumerates exactly the strings in $L \cap \Sigma^n$.

Theorem

The following are equivalent:

1. L is P-printable
2. for some k, $L \subseteq C[k \log n, n^k]$
3. for some k, $Ct(x) \leq k \log n$ for all $x \in L$

- Recall that a characteristic sequence of a set A, σ_A, is an infinite binary sequence whose i^{th} bit is 1 if the i^{th} string of Σ^* is in A. The finite sequence σ_A^n is the characteristic sequence of A through all of the strings of length up to n.

Theorem

A language A is in $\mathbf{P}_{/\mathbf{poly}}$ if and only if there is a constant c such that for all n:

$$CT(\sigma_A^n) \leq n^c$$

Theorem (Antunes-Fortnow-van Melkebeek)

The following are equivalent for all recursive languages L:

1. L is in $\mathbf{P}_{/\mathbf{poly}}$
2. There exists a set A and a constant k such that L is in $\mathbf{P}^A$ and

$$CT(\sigma_A^n) \leq K(\sigma_A^n) + n^k$$

for all n.

Other interesting applications

Theorem

A TM requires $\Omega(n^2)$ steps to recognize $L = \{xx^R : x \in \{0,1\}^*\}$.

Theorem

Let $n, r, s \in \mathbb{N}$ with $2\log n \leq r$, $s \leq \frac{n}{4}$ and s even. For each n there is a $n \times n$ matrix over $GF(2)$ such that every submatrix of s rows and $n - r$ columns has at least rank $s/2$.

Theorem

It requires $\Omega(n^{3/2}/\log n)$ time to deterministically simulate a linear-time 2-tape TM with one way input by a 1-tape TM with one-way input.

Håstad Switching Lemma

Let f be a t-CNF on n variables, ρ a random restriction $\in R_l$ and $\alpha = \frac{12tl}{n} \leq 1$. Then, the probability that $f|_\rho$ is an s-DNF is at least $1 - \alpha^s$.

- Other applications of the Incompressibility Method, including Tournaments, Ramsey Numbers, High-Probability properties of combinatorial objects, Kolmogorov Random Graphs, Compact Routing, Average-case analysis of Heapsort, Shellsort and LCS algos, Online CFL recognition.

Bibliography

- *An Introduction to Kolmogorov Complexity and Its Applications*, Ming Li and Paul Vitanyi. Springer, New York, 3rd edition, 2007
- *Information and Randomness: An Algorithmic Perspective*, Cristian Calude. Springer, Berlin, 2nd edition, 2002
- *Algorithmic Randomness and Complexity*, Rod Downey and Denis Hirschfeldt, Springer, Berlin, 2007
- *Kolmogorov Complexity and Computational Complexity*, Osamu Watanabe. Springer-Verlag, 1992
- *Algorithmic information theory*, Chaitin, G.J., IBM Journal of Research and Development, v.21, No. 4, 350359, 1977 *Kolmogorov Complexity and Computational Complexity*, Lance Fortnow
- *A Kolmogorov Complexity Proof of Håstad Switching Lemma: An Exposition*, Sophie Laplante

May the Force be with you!