CS154, Lecture 12: Kolmogorov Complexity: A Universal Theory of Data Compression
Rosencrantz & Guildenstern Are Dead (Tom Stoppard)
Rigged Lottery?

And the winning numbers are:
1, 2, 3, 4, 5, 6

But is this: “more random”?
The Church-Turing Thesis:

Everyone’s Intuitive Notion = Turing Machines of Algorithms

*This is not a theorem – it is a falsifiable scientific hypothesis.*

A Universal Theory of Computation
A Universal Theory of Information?

Can we quantify how much information is contained in a string?

A = 01010101010101010101010101010101

B = 110010011101110101101001011001011

Idea: The more we can “compress” a string, the less “information” it contains....
Information as Description

Thesis: The amount of information in a string $x$ is the length of the shortest description of $x$

How should we “describe” strings?

Use Turing machines with inputs!

Let $x \in \{0,1\}^*$

**Def:** A *description of* $x$ is a string $<M,w>$ such that $M$ on input $w$ halts with only $x$ on its tape.

**Def:** The *shortest description of* $x$, denoted as $d(x)$, is the lexicographically shortest string $<M,w>$ such that $M(w)$ halts with only $x$ on its tape.
A Specific Pairing Function

Theorem: There is a 1-1 computable function 
\( \langle, \rangle : \Sigma^* \times \Sigma^* \to \Sigma^* \) and computable functions 
\( \pi_1 \) and \( \pi_2 : \Sigma^* \to \Sigma^* \) such that:

\[ z = \langle M, w \rangle \text{ iff } \pi_1(z) = M \text{ and } \pi_2(z) = w \]

Define: \( \langle M, w \rangle := 0^{\left| M \right| + 1} M w \)

(Example: \( \langle 10110,101 \rangle = 00000110110101 \))

Note that \( |\langle M, w \rangle| = 2|M| + |w| + 1 \)
Kolmogorov Complexity (1960’s)

Definition: The shortest description of $x$, denoted as $d(x)$, is the lexicographically shortest string $\langle M, w \rangle$ such that $M(w)$ halts with only $x$ on its tape.

Definition: The Kolmogorov complexity of $x$, denoted as $K(x)$, is $|d(x)|$.

EXAMPLES??
Let’s first determine some properties of $K$. Examples will fall out of this.
A Simple Upper Bound

Theorem: There is a fixed $c$ so that for all $x$ in $\{0,1\}^*$
$$K(x) \leq |x| + c$$

“The amount of information in $x$ isn’t much more than $|x|$”

Proof: Define a TM $M =$ “On input $w$, halt.”
On any string $x$, $M(x)$ halts with $x$ on its tape.
Observe that $\langle M, x \rangle$ is a description of $x$.

Let $c = 2|M| + 1$
Then $K(x) \leq |\langle M, x \rangle| \leq 2|M| + |x| + 1 \leq |x| + c$
Theorem: There is a fixed $c$ so that for all $n \geq 2$, and all $x \in \{0,1\}^*$, 
$$K(x^n) \leq K(x) + c \log n$$

"The information in $x^n$ isn’t much more than that in $x$"

Proof: Define the TM 
$N = \text{“On input } \langle n, \langle M, w \rangle \rangle, \text{ let } x = M(w). \text{ Print } x \text{ for } n \text{ times.”} $

Let $\langle M, w \rangle$ be the shortest description of $x$. Then 
$$K(x^n) \leq K(\langle N, \langle n, \langle M, w \rangle \rangle \rangle) \leq 2|N| + d \log n + K(x) \leq c \log n + K(x)$$
for some constants $c$ and $d$
Repetitive Strings have Low K-Complexity

Theorem: There is a fixed $c$ so that for all $n \geq 2$, and all $x \in \{0,1\}^*$, $K(x^n) \leq K(x) + c \log n$

“The information in $x^n$ isn’t much more than that in $x$”

Recall: $A = 01010101010101010101010101010101$

For $w = (01)^n$, we have $K(w) \leq K(01) + c \log n$

So for all $n$, $K((01)^n) \leq d + c \log n$ for a fixed $c$ and $d$
Does The Computational Model Matter?

Turing machines are one “programming language.” If we use other programming languages, could we get significantly shorter descriptions?

An interpreter is a “semi-computable” function

\[ p : \Sigma^* \rightarrow \Sigma^* \]

Takes programs as input, and (may) print their outputs

Definition: Let \( x \in \{0,1\}^* \). The shortest description of \( x \) under \( p \), called \( d_p(x) \), is the lexicographically shortest string \( w \) for which \( p(w) = x \)

Definition: The \( K_p \) complexity of \( x \) is \( K_p(x) := |d_p(x)| \)
Theorem: For every interpreter \( p \), there is a fixed \( c \) so that for all \( x \in \{0,1\}^* \), \( K(x) \leq K_p(x) + c \)

Moral: Using another programming language would only change \( K(x) \) by some additive constant

Proof: Define \( M = \) “On \( w \), simulate \( p(w) \) and write its output to tape”
Then \( \langle M, d_p(x) \rangle \) is a description of \( x \), so
\[
K(x) \leq |\langle M, d_p(x) \rangle| \leq 2|M| + K_p(x) + 1 \leq c + K_p(x)
\]
There Exist Incompressible Strings

**Theorem**: For all \( n \), there is an \( x \in \{0,1\}^n \) such that \( K(x) \geq n \)

“There are incompressible strings of every length”

**Proof**: (Number of binary strings of length \( n \)) = \( 2^n \)

but (Number of descriptions of length \( \prec n \)) \( \leq \) (Number of binary strings of length \( \prec n \))

\( \leq 1 + 2 + 4 + \cdots + 2^{n-1} = 2^n - 1 \)

Therefore, there is at least one \( n \)-bit string \( x \) that does not have a description of length \( \prec n \)
Random Strings Are Incompressible!

**Theorem:** For all $n$ and $c \geq 1$,

$$\Pr_{x \in \{0,1\}^n}[K(x) \geq n-c] \geq 1 - 1/2^c$$

“Most strings are highly incompressible”

**Proof:**

(Number of binary strings of length $n$) = $2^n$

but (Number of descriptions of length $< n-c$) $\leq$ (Number of binary strings of length $< n-c$) = $2^{n-c} - 1$

Hence the probability that a random $x$ satisfies $K(x) < n-c$ is at most $(2^{n-c} - 1)/2^n < 1/2^c$. 
Kolmogorov Complexity: Try it!

Give short algorithms for generating the strings:

1. 01000110110000010100111001011101110000

2. 123581321345589144233377610987

3. 126241207205040403203628803628800
Kolmogorov Complexity: Try it!

Give short algorithms for generating the strings:

1. 01000110110000010100011100101110111000

2. 123581321345589144233377610987

3. 126241207205040403203628803628800
Kolmogorov Complexity: Try it!

Give short algorithms for generating the strings:

1. 01000110110000010100111001011101110000

2. 123581321345589144233377610987

3. 126241207205040403203628803628800
Kolmogorov Complexity: Try it!

Give short algorithms for generating the strings:

1. 0100011011000001010011001011101110000

2. 123581321345589144233377610987

3. 126241207205040403203628803628800

This seems hard to determine in general. Why?
Determining Compressibility?

Can an algorithm perform optimal compression? Can algorithms tell us if a given string is compressible?

\[ \text{COMPRESS} = \{(x,c) \mid K(x) \leq c\} \]

**Theorem:** COMPRESS is undecidable

**Idea:** If decidable, we could design an algorithm that prints the shortest incompressible string of length \( n \)

But such a string could then be succinctly described, by providing the algorithm code and \( n \) in binary!

**Berry Paradox:** “The smallest integer that cannot be defined in less than thirteen words.”
Determining Compressibility?

**COMPRESS** = \{ (x,c) | K(x) \leq c \}

**Theorem:** **COMPRESS** is undecidable!

**Proof:** Suppose it’s decidable. Consider the TM:

M = “On input x ∈ \{0,1\}^*, let N = 2^{|x|}.
For all y ∈ \{0,1\}^* in lexicographical order,
If (y,N) \notin COMPRESS then print y and halt.”

M(x) prints the shortest string y’ with K(y’) > 2^{|x|}.
\langle M,x \rangle is a description of y’, and |\langle M,x \rangle| \leq d + |x|.
So 2^{|x|} < K(y’) \leq d + |x|. CONTRADICTION for large x.
Yet Another Proof that \( A_{TM} \) is Undecidable

COMPRESS = \{ (x,c) | K(x) \leq c \}

**Theorem:** \( A_{TM} \) is undecidable.

**Proof:** Reduction from COMPRESS to \( A_{TM} \).

Given a pair \((x,c)\), our reduction constructs a TM:

\[
M_{x,c} = \text{On input } w,
\text{For all pairs } <M',w'> \text{ with } |<M',w'>| \leq c, \text{ simulate each } M' \text{ on } w' \text{ in parallel.}
\text{If some } M' \text{ halts and prints } x, \text{ then accept.}
\]

\[ K(x) \leq c \iff M_{x,c} \text{ accepts } \varepsilon \]
More Interesting Formal Systems

Define a formal system $F$ to be interesting if:

1. Any mathematical statement about computation can be (computably) described as a statement of $F$.
   Given $(M, w)$, there is a (computable) $S_{M,w}$ in $F$ such that $S_{M,w}$ is true in $F$ if and only if $M$ accepts $w$.

2. Proofs are “convincing” – a TM can check that a proof of a theorem is correct. This set is decidable: $\{(S, P) \mid P$ is a proof of $S$ in $F\}$

3. If $S$ is in $F$ and there is a proof of $S$ describable as a computation, then there’s a proof of $S$ in $F$.
   If $M$ accepts $w$, then there is a proof $P$ in $F$ of $S_{M,w}$.
The Unprovable Truth About K-Complexity

**Theorem:** For every interesting consistent $\mathcal{F}$, there is a $t$ s.t. for all $x$, “$K(x) > t$” is unprovable in $\mathcal{F}$

**Proof:** Define an $M$ that treats its input as an integer:

$$M(k) := \text{Search over all strings } x \text{ and proofs } P \text{ for a proof } P \in \mathcal{F} \text{ that } K(x) > k. \text{ Output } x \text{ if found}$$

Suppose $M(k)$ halts. It must print some output $x'$
Then $K(x') \leq K(<M,k>) \leq c + |k| \leq c + \log k$ for some $c$
Because $\mathcal{F}$ is consistent, $K(x') > k$ is true
But $k < c + \log k$ only holds for small enough $k$
If we choose $t$ to be greater than these $k$...
then $M(t)$ cannot halt, so “$K(x) > t$” has no proof!
Random Unprovable Truths

**Theorem**: For every interesting consistent $F$, there is a $t$ s.t. for all $x$, “$K(x) > t$” is unprovable in $F$.

For a randomly chosen $x$ of length $t+100$, “$K(x) > t$” is true with probability at least $1 - 1/2^{100}$.

We can (with high probability) *randomly generate* true statements in $F$ which have no proof in $F$.

For every interesting formal system $F$ there is always some finite integer (say, $t=10000$) so that you’ll never be able to prove in $F$ that a random 20000-bit string requires a 10000-bit program.