CS 154

Kolmogorov Complexity

New homework coming out today

The Church-Turing Thesis

Everyone's Intuitive Notion of Algorithms = Turing Machines

Turing machines are a "universal" notion of algorithm

Is there a Universal Notion 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 = Shortest way of describing that string

How should we "describe" strings?

Use Turing machines with inputs!

Definition: Let **x in {0,1}***. The **shortest description of x**, denoted as **d(x)**, is the **lexicographically shortest string <M,w>** s.t. M(w) halts with x on tape.

A Specific Pairing Function

Theorem. There is a 1-1 computable function $\langle , \rangle : \Sigma^* \times \Sigma^* \to \Sigma^*$ and computable functions π_1 and $\pi_2 : \Sigma^* \to \Sigma^*$ such that:

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

Let $Z(x_1 x_2 \dots x_k) = 0\, x_1\, 0\, x_2 \dots 0\, x_k\, 1$
Then we can define:

$$\langle M,w \rangle := Z(M)\, w$$

(Example: <10110,101> = 01000101001101)

Note that $|\langle M,w \rangle| = 2|M| + |w| + 1$

A Better Pairing Function

Let b(n) be the binary encoding of n
Again let $Z(x_1 x_2 \ldots x_k) = 0\, x_1\, 0\, x_2 \ldots 0\, x_k\, 1$

<M,w> := Z(b(|M|)) M w

Example: <10110,101>.
b(|10110|) = 101, so
<10110,101> = 010001110110101

We can still decode 10110 and 101 from this!

Now, |<M,w>| = 2 log(|M|) + |M| + |w| + 1

Kolmogorov Complexity

Definition: Let **x in {0,1}***. The **shortest description of x**, denoted as **d(x)**, is the **lexicographically shortest string <M,w>** s.t. M(w) halts with **x** on 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.

Kolmogorov Complexity

Theorem: There is a 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 M = "On input w, halt."
On any string x, M(x) halts with x on its tape.
This implies
$K(x) \leq |\langle M,x\rangle| \leq 2|M| + |x| + 1 \leq c + |x|$

Repetitive Strings have Low Info

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

"The information in **xx** isn't much more than that in **x**"

Proof: Let N = "On <M,w>, let s=M(w). Print ss."

Let <M,w> be the shortest description of x.
Then <N,<M,w>> is a description of xx

Therefore
$K(xx) \leq |\langle N,\langle M,x\rangle\rangle| \leq 2|N| + K(x) + 1 \leq c + K(x)$

Repetitive Strings have Low Info

Corollary: 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 = "On input <n,M,w>,
Let x = M(w). Print x for n times."

If <M,w> is the shortest description of x,
$K(x^n) \leq K(\langle N,\langle n,M,w\rangle\rangle) \leq 2|N| + d \log n + K(x)$
$\leq c \log n + K(x)$
for some c and d

Repetitive Strings have Low Info

Corollary: 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 = 0101010101010101010101010101010101

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

So, $K((01)^n) \leq O(\log n)$

Does The 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
$p : \Sigma^* \to \Sigma^*$
Takes programs as input, and prints their outputs

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

Definition: $K_p(x) := |d_p(x)|$.

Does The Model Matter?

Theorem: For every interpreter p, there is a c so that for all $x \in \{0,1\}^*$,
$K(x) \leq K_p(x) + c$

Moral: Using any other programming language would only change K(x) by some constant

Proof: Define M = "On w, output p(w)"
Then $\langle M, d_p(x) \rangle$ is a description of x, and
$K(x) \leq |\langle M, d_p(x) \rangle|$
$\leq 2|M| + K_p(x) + 1 \leq c + K_p(x)$

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
and (Number of descriptions of length < n)
$\leq$ (Number of binary strings of length < n)
$= 1 + 2 + 4 + \cdots + 2^{n-1} = 2^n - 1$

Therefore there's at least one n-bit string that does not have a description of length < n

Incompressible Strings

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

"Most strings are very incompressible"

Proof: (Number of binary strings of length n) = 2^n
and (Number of descriptions of length < n-c)
$\leq$ (Number of binary strings of length < n-c)
$= 2^{n-c} - 1$
So the probability that a *random* x satisfies
$K(x) < n-c$
is at most $(2^{n-c} - 1)/2^n < 1/2^c$.

Quiz

Give short algorithms for generating the strings:

1. **01000110110000010100111001011101110000**
2. **123581321345589144233377610987**
3. **126241207205040403203628803628800**

This seems hard to determine in general. Why? We'll give a formal answer in just one moment…

Determining Compressibility

Can an algorithm do optimal compression?
Can algorithms tell us if a string is compressible?
COMPRESS = $\{(x,c) \mid K(x) \leq c\}$

Theorem: COMPRESS is undecidable!

Intuition: If decidable, we can design an algorithm that prints the **shortest incompressible string of length n**
But such a string could be succinctly described, by giving the algorithm, 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:
M = "On input x $\in$ {0,1}*,
Interpret x as integer N. (Then, |x| $\leq$ log N)
For all y $\in$ {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') > N.
But <M,x> describes y', and |<M,x>| $\leq$ c + log N
So N < K(y') $\leq$ c + log N. CONTRADICTION!

Determining Compressibility

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

Theorem: A_{TM} is undecidable.

Proof: Reduction from COMPRESS to A_{TM}.
Given a pair (x,c), construct a TM $M_{x,c}$:

$M_{x,c}$ = Over all pairs <M',w> with |<M',w>| $\leq$ c,
Simulate each M' on w in parallel.
If some M' halts and prints x, then accept.

K(x) $\leq$ c if and only if $M_{x,c}$ accepts ε

More on Interesting Formal Systems

Recall a formal system $\mathcal{F}$ is *interesting* if:

1. Any mathematical statement describable in English can also be described within $\mathcal{F}$.
   *For all strings x and integers c, there is a $S_{x,c}$ in $\mathcal{F}$ that is equivalent to **"K(x) $\geq$ c"***
2. Proofs are convincing: it should be possible to check that a proof of a theorem is correct
   Given (S,P), it is decidable if P is a proof of S in $\mathcal{F}$
3. If there is a proof of S that's describable in English, then there's a proof describable in $\mathcal{F}$.
   *If **K(x) $\geq$ c** then there is a proof in $\mathcal{F}$ of $S_{x,c}$*

Random Unprovable Truths

Theorem: For every interesting consistent $\mathcal{F}$,
There is a t such that "K(x) > t" is unprovable in $\mathcal{F}$

Proof: Define a TM M:

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

Suppose M(k) halts with output x'
Then K(x') = K(<M,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 finitely many k
Choose t to be greater than all of these k…
Then *M(t) cannot halt, so "K(x) > t" has no proof!*

Random Unprovable Truths

Theorem: For every interesting consistent $\mathcal{F}$,
There is a t such that "K(x) > t" is unprovable in $\mathcal{F}$

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

We can *randomly generate* true statements in $\mathcal{F}$ which have no proof in $\mathcal{F}$ with high probability!

For every interesting formal system $\mathcal{F}$ there is always some finite constant (say, t=10000) so that you'll never prove in $\mathcal{F}$ that a random 20000-bit string requires a 10000-bit program!

Next Episode:

Complexity Theory!