CISC 876: Kolmogorov Complexity

Neil Conway

March 27, 2007

Outline

Outline

Complexity of Objects

Example

Which of these is more complex?

1. 1111111111111111
2. 1101010100011101

Complexity of Objects

Example

Which of these is more complex?

1. 1111111111111111
2. 1101010100011101

Intuition

- The first has a simple description: "print 1 16 times".
- There is no (obvious) description for the second string that is essentially shorter than listing its digits.

Complexity of Objects

Example

Which of these is more complex?

1. 1111111111111111
2. 1101010100011101

Intuition

- The first has a simple description: "print 1 16 times".
- There is no (obvious) description for the second string that is essentially shorter than listing its digits.
- Kolmogorov complexity formalizes this intuitive notion of complexity.

Complexity As Predictive Power

Solomonoff's Idea

Suppose a scientist takes a sequence of measurements: $x = \{0,1\}^*$. The scientist would like to formulate a hypothesis that predicts the future content of the sequence.

Among the infinite number of possible hypotheses, which should be preferred?

Complexity As Predictive Power

Solomonoff's Idea

Suppose a scientist takes a sequence of measurements: $x = \{0, 1\}^*$. The scientist would like to formulate a hypothesis that predicts the future content of the sequence.

Among the infinite number of possible hypotheses, which should be preferred?

Occam's Razor

Choose the simplest hypothesis that is consistent with the data

Algorithmic Information Theory

> *"Algorithmic information theory is the result of putting Shannon's information theory and Turing's computability theory into a cocktail shaker and shaking vigorously."*
> *—G. J. Chaitin*

Algorithmic Information Theory

> *"Algorithmic information theory is the result of putting Shannon's information theory and Turing's computability theory into a cocktail shaker and shaking vigorously."*
> *—G. J. Chaitin*

- AIT is a subfield of both information theory and computer science
- (Almost) simultaneously and independently developed by
  - 1962: introduced by R. J. Solomonoff as part of work on inductive inference
  - 1963: A. N. Kolmogorov
  - 1965: G. J. Chaitin (while an 18-year old undergraduate!)
- Also known as Kolmogorov-Chaitin complexity, descriptional complexity, program-size complexity, . . .

Outline

Definition

Definition

The Kolmogorov complexity of a string x is the length of the smallest program that outputs x, relative to some model of computation. That is,

$$C_f(x) = \min_p \{|p| : f(p) = x\}$$

for some computer f.

- Informally, $C(x)$ measures the information content, degree of redundancy, degree of structure, of x

Universality

Problem

$C_f(x)$ depends on both f and x. Can we measure the inherent information in x, independent of the choice of f?

Universality

Problem

$C_f(x)$ depends on both f and x. Can we measure the inherent information in x, independent of the choice of f?

Theorem (Invariance Theorem)

There exists a universal description method ψ_0, such that:

$$C_{\psi_0}(x) \leq C_\psi(x) + c$$

for some constant c that depends on ψ and ψ_0 (but not on x).

Proof Idea.

Follows from the existence of a universal Turing machine: accept a description of ψ and ψ's program for x □

Implications

Theorem

For all universal description methods f, g:

$$|C_f(x) - C_g(x)| \leq c$$

for some constant c that depends only on f and g.

- This is crucial to the usefulness of the complexity measure
- The universal description method does not necessarily give the shortest description of each object, but no other description method can improve on it by more than an additive constant
- We typically write $C(x) = C_{\psi_0}(x)$, use Turing machines as ψ_0, and limit our analysis to within an additive constant

Conditional Complexity

Definition

The conditional Kolmogorov complexity of a string x, relative to a string y and a model of computation f, is:

$$\begin{aligned} C_f(x|y) &= \min\{|p| : C_f(p, y) = x\} \\ C_f(x) &= C_f(x|\epsilon) \end{aligned}$$

- $C(x|y)$ is the size of the minimal program for x when started with input y
- $C(x:y) = C(x) - C(x|y)$ describes the information y contains about x
- When $C(x:y) = C(x)$, x and y are algorithmically independent

Simple Results

Upper Bound On $C(x)$

There is a constant c, such that for all x:

$$C(x) \leq |x| + c$$

(Proving a lower bound on $C(x)$ is not as straightforward.)

Structure and Complexity

For each constant k, there is a constant c such that for all x:

$$C(x^k) \leq C(x) + c$$

Incompressibility and Randomness

Definition

A string x is incompressible if

$$C(x) \geq |x|$$

- Maximal information content, no redundancy: algorithmically random
- Short programs encode patterns in non-random strings
- Algorithmic randomness is not identical to the intuitive concept of randomness
  - There is a short program for generating the digits of π, so they are highly "non-random"

Are There Incompressible Strings?

Theorem

For all n, there exists an incompressible string of length n

Proof.

There are 2^n strings of length n and fewer than 2^n descriptions that are shorter than n:

$$\sum_{i=0}^{n-1} 2^i = 2^n - 1 < 2^n$$

□

Incompressibility Theorem

We can extend the previous counting argument to show that the vast majority of strings are mostly incompressible

Definition

A string x is c-incompressible if $C(x) \geq |x| - c$, for some constant c.

Theorem

The number of strings of length n that are c-incompressible is at least

$$2^n - 2^{n-c+1} + 1$$

Example

For $c = 10$:

The fraction of all strings of length n with complexity less than $n - 10$ is smaller than:

$$\frac{2^{n-11+1}}{2^n} = \frac{1}{1024}$$

Consequences

Fact

The probability that an infinite sequence obtained by independent tosses of a fair coin is algorithmically random is 1.

Fact

The minimal program for any string is algorithmically random.

Noncomputability Theorem

Theorem

$C(x)$ is not a computable function.

Proof.

Will be presented shortly. □

Conclusions

- Given any concrete string, we cannot show that it is random
  - Apparent randomness may be the result of a hidden structure
  - Wolfram's conjecture: much/all apparent physical randomness is ultimately the result of structure
- "Almost all" strings are algorithmically random, but we cannot exhibit any particular string that is random
- There are relatively few short programs, and relatively few objects of low complexity

Outline

Additive Complexity

Theorem

$$C(x, y) = C(x) + C(y) + O(\log(min(C(x), C(y))))$$

Proof Idea.

1. ($\leq$): Construct a TM that accepts descriptions (programs) for x, y, and a way to distinguish them
   - The length of the shorter input
2. ($\geq$): It can be shown that we cannot do better than this for all but finitely many x, y

□

Consequences

- This is unfortunate: we would like K-complexity to be subadditive
  - $C(x) + C(y)$ should bound $C(x, y)$ from above
- We would also like to combine subprograms by simple concatenation

Self-Delimiting Strings

Definition

A string is self-delimiting if it contains its own length.

Self-Delimiting Strings

Definition

A string is self-delimiting if it contains its own length.

Procedure

- Prepend the string's length to the string
- Problem: how can we distinguish the end of the length from the start of the string itself?

Self-Delimiting Strings

Definition

A string is self-delimiting if it contains its own length.

Procedure

- Prepend the string's length to the string
- Problem: how can we distinguish the end of the length from the start of the string itself?
- Solution: duplicate every bit of the length, then mark the end of the length with 01 or 10
- A binary string of length n can be encoded in self-delimiting form in $n + 2\log n$ bits

Prefix Complexity

- Most modern work on Kolmogorov complexity actually uses prefix complexity, a variant formulated by L. A. Levin (1974)
- $K(x)$ is the size of the minimal self-delimiting program that outputs x; $K(x)$ is subadditive
- No self-delimiting string is the prefix of another
- Various other helpful theoretical properties

Summary of Results

- upper bounds: $K(x) \leq l(x) + 2\log l(x)$, $K(x|l(x)) = l(x)$

Summary of Results

- upper bounds: $K(x) \leq l(x) + 2 \log l(x)$, $K(x|l(x)) = l(x)$
- extra information: $K(x|y) \leq K(x) \leq K(x, y)$

Summary of Results

- upper bounds: $K(x) \leq l(x) + 2 \log l(x)$, $K(x|l(x)) = l(x)$
- extra information: $K(x|y) \leq K(x) \leq K(x, y)$
- subadditive: $K(x, y) \leq K(x) + K(y)$

Summary of Results

- upper bounds: $K(x) \leq l(x) + 2 \log l(x)$, $K(x|l(x)) = l(x)$
- extra information: $K(x|y) \leq K(x) \leq K(x, y)$
- subadditive: $K(x, y) \leq K(x) + K(y)$
- symmetry of information: $K(x, y) \leq K(y, x)$

Summary of Results

- upper bounds: $K(x) \leq l(x) + 2\log l(x)$, $K(x|l(x)) = l(x)$
- extra information: $K(x|y) \leq K(x) \leq K(x,y)$
- subadditive: $K(x,y) \leq K(x) + K(y)$
- symmetry of information: $K(x,y) \leq K(y,x)$
- lower bound: $K(x) \geq l(x)$ for "almost all" x

Resource-Bounded Kolmogorov Complexity

Definition

Intuitively, a string has high logical depth if it is "superficially random, but subtly redundant": the string has low complexity, but only for a computational model with access to a lot of resources

Resource-Bounded Kolmogorov Complexity

Definition

Intuitively, a string has high logical depth if it is "superficially random, but subtly redundant": the string has low complexity, but only for a computational model with access to a lot of resources

- We can consider the complexity of a string, relative to a computational model with bounded space or time resources
- Typically harder to prove results than with unbounded K-complexity

Invariance Theorem with Resource Bounds

Theorem (Invariance Theorem)

There exists a universal description method ψ_0, such that for all other description methods ψ we have a constant c such that:

$$C_{\psi_0}^{ct \log n, cs}(x) = C_{\psi}^{t,s}(x) + c$$

Problem

Considerably weaker Invariance Theorem: multiplicative constant factor in space complexity, multiplicative logarithmic factor in time complexity.

Relation To Other Fields

- Shannon's information theory
  - The information required to select an element from a previously agreed-upon set of alternatives

Relation To Other Fields

- Shannon's information theory
  - The information required to select an element from a previously agreed-upon set of alternatives
- Minimum Description Length (MDL)
  - Place limitations on the computation model so the MDL of a string is computable
  - Closer to learning theory and Solomonoff's work on inductive inference

Relation To Other Fields

- Shannon's information theory
  - The information required to select an element from a previously agreed-upon set of alternatives
- Minimum Description Length (MDL)
  - Place limitations on the computation model so the MDL of a string is computable
  - Closer to learning theory and Solomonoff's work on inductive inference
- Circuit complexity
  - Kolmogorov complexity considers Turing machines rather than Boolean circuits

Outline

Incompressibility Method

A general-purpose method for formal proofs; often an alternative to counting arguments or probabilistic arguments

Typical Proof Structure.

To show that "almost all" the objects in a given class have a certain property:

1. Choose a random object from the class
2. This object is incompressible, with probability 1
3. Prove that the property holds for the object
    1. Assume that the property does not hold
    2. Show that we can use the property to compress the object, yielding a contradiction

□

Simple Example

Theorem

$L = \{0^k 1^k : k \geq 1\}$ *is not regular.*

Proof Idea.

1. Choose k such that k is Kolmogorov-random

□

Simple Example

Theorem

$L = \{0^k 1^k : k \geq 1\}$ *is not regular.*

Proof Idea.

1. Choose k such that k is Kolmogorov-random
2. Assume that $0^k 1^k$ is a regular language, and is accepted by some finite automaton A

□

Simple Example

Theorem

$L = \{0^k 1^k : k \geq 1\}$ *is not regular.*

Proof Idea.

1. Choose k such that k is Kolmogorov-random
2. Assume that $0^k 1^k$ is a regular language, and is accepted by some finite automaton A
3. After input 0^k, A is in state q

□

Simple Example

Theorem

$L = \{0^k 1^k : k \geq 1\}$ *is not regular.*

Proof Idea.

1. Choose k such that k is Kolmogorov-random
2. Assume that $0^k 1^k$ is a regular language, and is accepted by some finite automaton A
3. After input 0^k, A is in state q
4. A and q form a concise description of k: running A from state q accepts only on an input of k consecutive 1s

□

Simple Example

Theorem

$L = \{0^k 1^k : k \geq 1\}$ *is not regular.*

Proof Idea.

1. Choose k such that k is Kolmogorov-random
2. Assume that $0^k 1^k$ is a regular language, and is accepted by some finite automaton A
3. After input 0^k, A is in state q
4. A and q form a concise description of k: running A from state q accepts only on an input of k consecutive 1s
5. This contradicts the assumption that k is incompressible

□

Properties of Formal Languages

- Pumping lemmas are the standard tool for showing that a language is not in REG, DCFL, CFL, ...
- Kolmogorov complexity provides an alternative way to characterize membership in these classes
  - Can prove both regularity and non-regularity

K-Complexity Analog to the Pumping Lemma for REG

Lemma (Kolmogorov-Complexity-Regularity (KCR))

Let L be a regular language. Then for some c depending only on L and for each x, if y is the nth string in lexicographical order $L_x = \{y : xy \in L\}$, then $K(y) \leq K(n) + c$.

K-Complexity Analog to the Pumping Lemma for REG

Lemma (Kolmogorov-Complexity-Regularity (KCR))

Let L be a regular language. Then for some c depending only on L and for each x, if y is the nth string in lexicographical order $L_x = \{y : xy \in L\}$, then $K(y) \leq K(n) + c$.

Proof.

Any string y such that $xy \in L$, can be described by:

1. the description of the FA that accepts L
2. the state of the FA after processing x
3. the number n

□

Example of KCR Lemma

Fact

$L = \{0^n : n \text{ is prime } \}$ *is not regular.*

Example of KCR Lemma

Fact

$L = \{0^n : n \text{ is prime }\}$ *is not regular.*

Proof.

Assume that L is regular. Set $xy = 0^p$ and $x = 0^{p'}$, where p is the k'th prime and p' is the $(k-1)$th prime. It follows that $y = 0^{p-p'}$, $n = 1$, and $K(p - p') = O(1)$.

Example of KCR Lemma

Fact

$L = \{0^n : n \text{ is prime }\}$ *is not regular.*

Proof.

Assume that L is regular. Set $xy = 0^p$ and $x = 0^{p'}$, where p is the k'th prime and p' is the $(k-1)$th prime. It follows that $y = 0^{p-p'}$, $n = 1$, and $K(p - p') = O(1)$.

This is a contradiction: the difference between consecutive primes rises unbounded, so there are an unbounded number of integers with $O(1)$ descriptions. □

Other Applications of the Incompressibility Method

- Average-case complexity analysis
  - Avoids the need to explicitly model the probability distribution of inputs
  - E.g. heapsort
- Lower bounds analysis for problems
- Properties of random graphs
- Typically yields proofs that are shorter and more elegant than alternative techniques

Historical Context: Formal Axiomatic Systems

- Turn of the 20th century: what constitutes a valid proof?
- David Hilbert's program: can we formalize mathematics?

Hilbert's 2nd Problem (1900)

Construct a single formal axiomatic system that contains all true arithmetical statements over the natural numbers:

- A finite number of axioms, and a deterministic inference procedure
- Consistent: no contradictions can be derived from the axioms
- Complete: all true statements can be derived from the axioms

Gödel's Incompleteness Theorems

Theorem (1st Incompleteness Theorem)

Any computably enumerable, consistent formal axiomatic system containing elementary arithmetic is incomplete: there exist true, but unprovable (within the system) statements.

Gödel's Incompleteness Theorems

Theorem (1st Incompleteness Theorem)

Any computably enumerable, consistent formal axiomatic system containing elementary arithmetic is incomplete: there exist true, but unprovable (within the system) statements.

Theorem (2nd Incompleteness Theorem)

The consistency of a formal axiomatic system that contains arithmetic cannot be proven within the system.

Consequences

- For each true, unprovable statement, we can "solve" the problem by adding a new axiom to the system
  - There are an infinity of such unprovable statements, so we never achieve completeness

Consequences

- For each true, unprovable statement, we can "solve" the problem by adding a new axiom to the system
  - There are an infinity of such unprovable statements, so we never achieve completeness
- Hilbert's program is not achievable: any single axiomatization of number theory cannot capture all number-theoretical truths
- However, does not invalidate formalism itself: many formal models are now necessary rather than a single one

Consequences

- For each true, unprovable statement, we can “solve” the problem by adding a new axiom to the system
  - There are an infinity of such unprovable statements, so we never achieve completeness
- Hilbert's program is not achievable: any single axiomatization of number theory cannot capture all number-theoretical truths
- However, does not invalidate formalism itself: many formal models are now necessary rather than a single one
- “Provability is a weaker notion than truth.” —Douglas Hofstadter
- . . . and much more philosophical speculation in the same vein

Connection to Kolmogorov Complexity

- Gödel's proof relies on an ingenious technique: in any formal system that contains arithmetic, we can construct a true theorem in the formal system that encodes the assertion "This theorem is not provable within the system"
  - Neat, but an artificial construction
- How widespread are these true, unprovable statements?
- K-complexity allows a simple proof of Gödel's incompleteness results that sheds more light on the power of formal systems

Complexity of a Formal System

Definition

The complexity of a formal system is the size of the minimal program that lists all the theorems in the system.

- Equivalently, a formal system's complexity is the size of the minimal encoding of the alphabet, axioms, and inference procedure

Solving the Halting Problem

Theorem

A formal system of complexity n can solve the Halting Problem for programs smaller than n bits.

Proof.

The system can contain at most n bits of axioms. This is enough space to specify the number of n-bit programs that halt, or equivalently to identify the halting program with the longest runtime. □

Corollary

A finite formal axiomatic system can only prove finitely many statements of the form $C(x) > m$.

Corollary: K(x) Is Uncomputable

Theorem

No formal system of complexity n can prove that an object x has $K(x) > n$.

Proof Idea.

If the formal system can prove that x has complexity $n' = K(x)$, this encodes a description of x in n bits. Since $n' > n$, we have a contradiction. □

AIT Restatement of Incompleteness

Theorem

There are true but unprovable statements in any consistent formal axiomatic system of finite size.

Proof Idea.

Follows from the earlier two results. □

Outline

Summary

- Kolmogorov complexity measures the absolute information content of a string, to within an additive constant
- The uncomputability of K-complexity is an obstacle
- The incompressibility method is a useful (advanced) proof technique
- AIT allows a simple proof of the Incompleteness theorem, as well as more insight into the nature of formal axiomatic systems