

A little Information Theory

”[Chaitin’s Omega] bids fair to hold
the mysteries of the universe”

M. Gardner

WH

January 11, 2006

Three people:

Claude E. Shannon (1916 – 2001)

The information content of a message consists simply of the 1s and 0s it takes to transmit it.

Andrey Nikolaevich Kolmogorov (1903 – 1987)

The complexity of a string is the length of the shortest program able to generate it.

Gregory J. Chaitin (1947 –) <http://cs.umaine.edu/~chaitin/>

If a theorem contains more information than a given set of axioms then it cannot be derived from them.

Information theory begins...

A Mathematical Theory of Communication

1949, Claude E. Shannon, [Sha48].

Messages, of any sort, are simply sequences of bits.

Kolmogorov-Chaitin Complexity

Definition: The *algorithmic information content* of a binary string is the size of the smallest program that can generate it.

Corollary: A string is *random* when it cannot be generated by any program smaller than itself.

Note: it is then a contradiction in terms to say that a program could generate an infinite random bit-stream.

Warming up...

Theorem: There are infinitely many prime numbers.

Proof: Imagine the contrary, that the primes numbers are P_1, P_2, \dots, P_m . Lemma: any number can be uniquely represented as a product of primes. So, $\forall n. \exists a_1, \dots, a_m. n = P_1^{a_1} P_2^{a_2} \dots P_m^{a_m}$, i.e. any number can be represented as a vector $V(N) = a \in \mathbb{N}^m$. Let $I(N)$ equal the *information content* of N . The information content of a vector representing some number N is *at most* $m \log \log N$, hence

$$I(N) \leq m \log \log N \quad (1)$$

But let N be large and algorithmically random, and we find

$$I(N) = \log N \leq m \log \log N \quad (2)$$

A contradiction!

□

Slightly more advanced...

A form of Gödel's Incompleteness Theorem

- By counting-arguments, *most* numbers are *random*
- However, given any algebraic framework – some axioms, rules-of-inferences etc – one can only *prove* that finitely many of them are random:
- A formal system written in n bits cannot prove the randomness of a number greater than n bits in length.

Proof:

Imagine the contrary, a formal system of n bits that can prove the randomness of a random number $m \gg n$ bits – of, say, the first random number of m bits.

Write a program based on our formal system, that takes each m bit number in turn, checks whether it is random, and if so outputs it and halts.

Then, we would have a program of fewer than m bits which can write a random number of m bits – a contradiction! \square

Note: the typical interpretation of Gödel's Incompleteness Theorem is that it is an artefact of self-reference (“This statement is unprovable”)

Not here! No self-reference needed.

Gregory Chaitin

- Website, <http://www.cs.auckland.ac.nz/CDMTCS/chaitin/>
- Excellent section titled, *Understandable Papers*

Chaitin's Omega

- *Strongly* random:
- There is no possible strategy for guessing what its next digit might be.
- Ω = the probability that a random sequence of digits, fed into a Universal Turing Machine, will cause a halting computation
- If a few thousand of its digits were known, we could answer most of the problems open in mathematics.

Technical:

Assume that no program (sequence of bits) b prefixes another – i.e. if b is a program, and $e = bd$ then e is not a program.

Not a big deal: make, say, “000” an end-of-program token. The set of programs is,

$$B = \{b \in \{0, 1\}^*000 \mid \#_{000}b = 1\} \quad (3)$$

and then,

$$\Omega = \sum_{\substack{b \in B \\ b \downarrow}} 2^{-|b|} \quad (4)$$

If no programs halt, $\Omega = 0$; if all programs halt then $\Omega = 1/8$ (in this scheme – in others $\Omega = 1$, but they are more technical). Of course, some programs halt, and others do not, and that is our problem...

Technical (part II):

We're interested in finite initial parts of Ω ,

$$\Omega = b_1 \frac{1}{2} + b_2 \frac{1}{2^2} + b_3 \frac{1}{2^3} + \dots \quad (5)$$

$$\Omega_n = b_1 \frac{1}{2} + b_2 \frac{1}{2^2} + b_3 \frac{1}{2^3} + \dots + b_n \frac{1}{2^n} \quad (6)$$

Note well that,

$$\Omega_n < \Omega < \Omega_n + \frac{1}{2^n} \quad (7)$$

Now, we're going to use Ω_n , for a *small value* of n , to solve one of the most celebrated open problems in pure mathematics...

For example...

Goldbach's Conjecture: (actually, Euler's Conjecture) Every even number greater than 4 is equal to the sum of two primes.

An open problem since 1742. If we had even a small amount of Ω , we could solve it.

Program G:

Take each natural number n in turn;

Is n the sum of two primes?

If yes, continue; if no, halt.

Then, Goldbach's Conjecture is true if and only if **G** does not halt.

Let \mathbf{G} be m bits long. We need $\Omega_m \dots$

Procedure:

For $n = 1, 2, 3, \dots$

Start running all problems of length n

Whenever a program of length i halts,

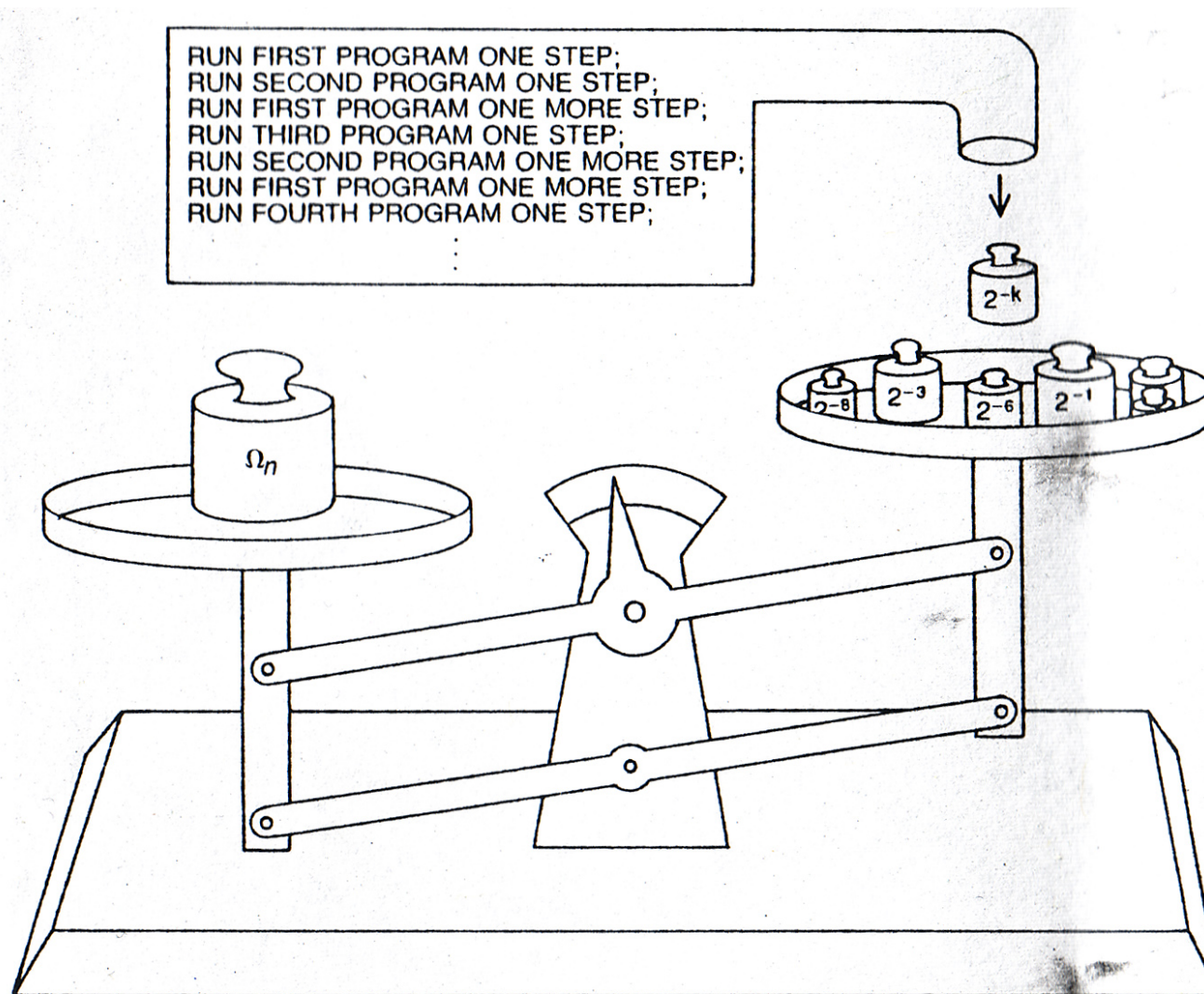
add 2^{-i} to the pan

If the pan is heavier than Ω_m ,

output: “ \mathbf{G} does not halt”

Then, *eventually*, either \mathbf{G} halts, or the program above tells us that it will never do so. Done!

The point is: once we have accumulated Ω_m in the bucket, \mathbf{G} cannot possibly be a halting program, otherwise $\Omega \leq \Omega_m + \frac{1}{2^m}$. \square



Using the first n bits of Ω to solve the halting problem for all programs of n bits or fewer

References

- [Gar79] Gardner. Mathematical games: The random number omega bids fair to hold the mysteries of the universe. *SIAM: Scientific American*, 241, 1979.
- [Sha48] C. E. Shannon. A mathematical theory of communication. *Bell Sys. Tech. J.*, 27:379–423, 623–656, 1948.