The Halting Probability via Wang Tiles

Gregory Chaitin*

Abstract

Using work of Hao Wang, we exhibit a tiling characterization of the bits of the halting probability Ω.

Algorithmic information theory [2] shows that pure mathematics is infinitely complex and contains irreducible complexity. The canonical example of such irreducible complexity is the infinite sequence of bits in the base-two expansion of the halting probability Ω. The halting probability is defined by taking the following summation

$$0 \quad < \quad \Omega \quad = \sum_{U(p) \text{ halts}} 2^{-|p|} \quad < \quad 1$$

over all the self-delimiting programs p that halt when run on a suitably defined universal Turing machine U. Here $|p|$ denotes the size in bits of the program p. The value of Ω depends on the choice of U, but its surprising properties do not.

The numerical value of Ω is *maximally unknowable* in the following precise sense. You need an N-bit theory in order to be able to determine N bits of Ω [5]. Nevertheless, Ω has a kind of *diophantine reality,* because there is a diophantine equation with a parameter k that has finitely or infinitely many solutions depending on whether the kth bit of Ω is respectively 0 or 1 [2].

*IBM T. J. Watson Research Center, P. O. Box 218, Yorktown Heights, NY 10598, U.S.A., *chaitin@us.ibm.com.*

More recently, Ord and Kieu [3] have shown that there is also a diophantine equation with a parameter k that has an even or odd number of solutions depending on whether the kth bit of Ω is respectively 0 or 1.

In [4] we show that as well as "diophantine reality," Ω also possesses a kind of *algebraic reality,* because there is an algebraic problem with a parameter i which yields the infinite sequence of bits b_i in the binary expansion of Ω:

$$\Omega = \sum_{i=1,2,3,\ldots} b_i \times 2^{-i}.$$

The purpose of this note is to discuss the fact that the bits of Ω can also be dressed up as facts about infinite tilings using a specific fixed set W of Wang tiles (also referred to as Wang dominoes) with a certain distinguished subset $D \subset W$. In particular, consider the infinite half-plane extending to the right. Given i, we can place a vertical strip of Wang tiles on the left-most column of this half-plane, in such a manner that this determines a single unique tiling of the infinite half-plane using the Wang tiles W. This unique tiling will contain finitely or infinitely many tiles from the distinguished subset D depending on whether b_i is respectively 0 or 1.

We proceed to the proof.

First of all, note that one can calculate better and better lower bounds on Ω, for example, by using the simple LISP function given in [6, pp. 65–69]. This works because Ω is the limit of Ω_n defined as follows:

$$\Omega_n = \sum_{|p| \le n \text{ and } U(p) \text{ halts in } \le n \text{ steps}} 2^{-|p|}.$$

As n tends to infinity, Ω_n tends to Ω, and from some point on each bit of Ω_n will remain correct, since Ω is irrational.[1] In other words, as n tends to infinity, the values of individual bits of Ω_n will fluctuate but eventually settle down to the correct values.

Wang tiles have an entire chapter devoted to them in the standard reference Grünbaum and Shephard [1]. Nevertheless [1] does not show how to simulate a Turing machine using Wang tiles. The beauty of Wang tiles is that there is an easy way to do such simulations, so a proof of this fact is included here.

[1] I.e., this limiting process cannot give us .3659999... instead of .3660000... because then Ω would be a rational number and would therefore not be irreducibly complex.

Wang tiles consist of identical squares divided into 4 triangular pieces by their two diagonals:

$$\boxtimes$$

We shall refer to the 4 pieces as left, right and top and bottom quadrants. Each of these quadrants has a particular color, and we are given a finite set W of these tiles, each with a specific choice of colors for each of its four quadrants. The rules for tiling with these Wang tiles W is that adjacent tiles must have abutting edges with matching colors, one can use as many copies of each tile as one wishes, one cannot rotate or reflect tiles in W, and they must all fit together in a regular square grid.

It is easy to simulate a Turing machine with a single two-way infinite tape and a single read-write head using Wang tiles. The strategy is to make the successive contents of the Turing machine tape into successive vertical columns of Wang tiles, starting with the initial tape contents in the left-most vertical column of the half-plane. So increasing time in the Turing machine computation corresponds to moving rightwards in the tiling. In other words, the tiling is a kind of "spacetime" diagram of the computation.

Furthermore, the internal state of the Turing machine moves on the tape together with the read-write head. In other words, we pretend the internal state **is** the read-write head. So the internal state will move across the spacetime diagram of the computation following the motion of the read-write head.

How are the Wang tiles W_T corresponding to a specific Turing machine T colored?

The first step is to choose our set of colors. The colors for the Wang tiles correspond to each of the following: quiescent (BLACK), Turing machine tape symbols, Turing machine internal states, and (tape symbol, internal state) pairs. There is a unique color for each possibility in this list.

The next step is to indicate the choice of colors for each tile in W_T. To do this I will employ diagrams of this form:

	color of top quadrant ↓	
color of left quadrant →	WANG TILE	→ color of right quadrant
	↑ color of bottom quadrant	

The directions of the arrows are intended to indicate in which direction information is flowing in the spacetime diagram.

Now let's use these diagrams to exhibit the tiles that we need to simulate a computation.

The first step is to ensure that the inactive tape cells of the Turing machine are simulated. These are the ones far from the read-write head. The following tiles will ensure this:

quiescent
↓
tape symbol → WANG TILE → same tape symbol
↑
quiescent

Now let's take a look at the active tape square, the one with the read-write head. The simplest case is if the read-write head doesn't move. But the tape symbol and internal state can change, depending on the state and the tape symbol that was read:

quiescent
↑
(tape symbol, state) → WANG TILE → (new tape symbol, new state)
↓
quiescent

If the read-write head moves, then the active square becomes inactive. This is done by choosing one of the following two tiles, depending on the direction of motion of the read-write head:

new state
↑
(tape symbol, state) → WANG TILE → new tape symbol
↓
quiescent

quiescent
↑
(tape symbol, state) → WANG TILE → new tape symbol
↓
new state

And here is how we activate inactive tape squares by moving the read-write head to them:

new internal state
↓
tape symbol → WANG TILE → (same tape symbol, new internal state)
↑
quiescent

quiescent
↓
tape symbol → WANG TILE → (same tape symbol, new internal state)
↑
new internal state

Up to here, we have just shown how to use Wang tiles to simulate a Turing machine. Now let's tailor the construction to represent the bits of Ω. The first step is to produce the finite set of Wang tiles corresponding to the following Turing machine: Given n 1's on its initial tape, with everything else blank and the read-write head in the initial state on the top (first) 1, it calculates successive approximations to b_n, more precisely, it calculates the nth bit of Ω_k for $k = 1, 2, 3, \ldots$ This is an unending computation, and certain distinguished internal states correspond to outputting a 1 bit, i.e., to the fact that the nth bit of Ω_k is 1. All the Wang tiles which have these special "output 1" internal states are placed in the distinguished subset D of our tile set W.

If the nth bit of Ω is 1, then for all sufficiently large k, the nth bit of Ω_k will be 1, and the Turing machine will visit infinitely many "output 1" states, and there will be infinitely many tiles in D in the unique tiling that is a spacetime diagram of this unending computation. Contrariwise, if the nth bit of Ω is 0, then for all sufficiently large k, the nth bit of Ω_k will be 0, and the Turing machine will not visit infinitely many "output 1" states, and there will be only finitely many tiles in D in the unique tiling that is a spacetime diagram of this unending computation.

This completes the proof.

Using the technique of [3], which is also presented in [5], we can modify this construction somewhat. As before, the bits of Ω are dressed up as facts about infinite tilings using a specific fixed set W of Wang tiles with a certain distinguished subset $D \subset W$. Consider the infinite half-plane extending to the right. Given i, we can place a vertical strip of Wang tiles on the left-most column of this half-plane, in such a manner that this determines a single unique tiling of the infinite half-plane using the Wang tiles W. In the alternative version of our construction based on the technique of Ord and Kieu [3], this unique tiling will **always** contain **finitely** many tiles from the distinguished subset D. However the number of tiles in D in this unique infinite tiling depending on i will be **even** or **odd** depending on whether the ith bit of Ω, b_i, is respectively 0 or 1.

For the philosophical significance of Ω, see [7].

References

[1] B. Grünbaum, G. C. Shephard, *Tilings and Patterns,* W. H. Freeman, 1987, Chapter 11.

[2] G. Chaitin, *Algorithmic Information Theory,* Cambridge University Press, 1987.

[3] T. Ord, T. D. Kieu, "On the existence of a new family of diophantine equations for Ω," *Fundamenta Informaticae,* Vol. 56 (2003), pp. 273–284.

[4] G. Chaitin, "An algebraic characterization of the halting probability," *Fundamenta Informaticae,* Vol. 79 (2007), pp. 17–23.

[5] G. Chaitin, *Meta Math!,* Pantheon, 2005.

[6] G. Chaitin, *The Limits of Mathematics,* Springer-Verlag, 1998.

[7] G. Chaitin, *Thinking about Gödel & Turing: Essays on Complexity, 1970–2007,* World Scientific, 2007.