TURING'S FIXED POINT THEOREM

Aidan Rocke
aidanrocke@gmail.com

February 28, 2021

"The view that machines cannot give rise to surprises is due, I believe, to a fallacy to which philosophers and mathematicians are particularly subject. This is the assumption that as soon as a fact is presented to a mind all consequences of that fact spring into the mind simultaneously with it. It is a very useful assumption under many circumstances, but one too easily forgets that it is false."-Alan Turing

1 Introduction

It is worth noting that any loop involves a sequence of functions $\{f_i\}_{i=1}^n$ composed with each other $f_{i+1} \circ f_i$ with the restriction that:

$$\forall i, f_{i+1} \equiv f_i \Rightarrow f^2 = f_{i+1} \circ f_i \tag{1}$$

Now, it is natural to ask whether it is possible to define a halting criterion G for any computable function f such that $\forall x \in Dom(f)$:

$$\forall n \in \mathbb{N}, G \circ f^n \circ x = 1 \Rightarrow \textbf{program halts} \tag{2}$$

$$\forall n \in \mathbb{N}, G \circ f^n \circ x = 0 \Rightarrow f \circ f^n \circ x \tag{3}$$

and most importantly we would like to guarantee that G eventually halts the process f:

$$\exists n \in \mathbb{N}, G \circ f^n \circ x = 1 \Rightarrow \textbf{program halts} \tag{4}$$

which is to say that the loop eventually terminates. What Turing found is that there are some functions f that are a fixed point of f i.e. there are some loops which never terminate and more importantly it is generally impossible to know whether a function f belongs to the class of functions that halt on all its inputs or not. From this we may deduce that within the realm of computable functions, a function G that serves as a general halting criterion for all computable functions does not exist.

From a dynamical systems perspective Turing's formulation of the halting problem has two interesting features. First, it clearly shows that that there is an intrinsic directionality in the information processing behaviour of Turing Machines. Second, its expression as a fixed point theorem presciently anticipates the development of Chaos theory for deterministic dynamical systems.

By formulating Turing's theorem in this manner we also manage to address the misplaced criticism from the computational biology community that biological systems are dynamical systems and not computers.

2 Turing's theorem:

2.1 Statement of the theorem:

Let H be a computable function which takes two inputs $a, b \in \{0,1\}^*$ and always returns 0(i.e. halts) or 1(i.e. an infinite loop):

$$H : \{0,1\}^* \times \{0,1\}^* \to \{0,1\} \tag{5}$$

where $\{0,1\}^*$ is the space of all binary strings, which naturally includes binary encodings of computable functions.

Then there exists a function $P \in \{0,1\}^*$ such that $P \circ P$ halts if and only if $H(P,P)$ is an infinite loop.

2.2 Proof:

Let's consider the function:

$$\forall x \in \{0,1\}^*, P \circ x = \textbf{Bool} \circ \{H(x,x) = 0\} \tag{6}$$

Given that $H(P,P)$ is either one or zero we have two cases:

$$H(P,P) = 0 \Rightarrow P \circ P = 1 \tag{7}$$

$$H(P,P) = 1 \Rightarrow P \circ P = 0 \tag{8}$$

However, in boolean arithmetic both cases reduce to:

$$H(P,P) + P \circ P = 1 \tag{9}$$

so P enters an infinite loop either way or to be mathematically precise, P is a fixed point of the computable function P.

3 Turing's theorem from a dynamical systems perspective

From a dynamical systems perspective, the stopping criterion H is equivalent to stating that for any deterministic transformation f operating on a state-space X:

$$\forall x_0 \in X, x_{n+1} = f \circ x_n \tag{10}$$

and we may decompose the state-space into $X = \Lambda \cup X \setminus \Lambda$ such that the question of whether Λ contains all the attractors of f is a decidable problem.

Given the space of computable functions F, this decidable problem may be formulated in terms of the halting function H so we have:

$$\forall f \in F, H \circ f = \textbf{Bool} \circ \{\forall x \in Dom(f) \exists n \in \mathbb{N}, f^n \circ x \in \Lambda\} \in 0,1 \tag{11}$$

such that H is defined $\forall f \in F$, i.e. H is a total function. Turing showed that such a function H does not exist.

This is akin to asking whether given $x_0 \in Dom(f)$, we may predict the destiny of x_0 knowing that:

$$x_{n+1} = f \circ x_n \tag{12}$$

and assuming that we know everything that can be known about f. Turing's theorem shows that that this is generally impossible; Turing Machines are constrained by a principle of limited omniscience.

In fact, this result anticipated in a profound sense the development of Chaos theory and the Butterfly effect in particular. So we may say that Turing's fixed point theorem is a profound result in Chaos theory and has important consequences for epistemology in general.

4 Discussion

This is a very important result in the natural sciences because it shows that there is a fundamental asymmetry between mathematically-defined computational processes which are not guaranteed to halt and *physical processes* which are guaranteed to halt due to energy constraints. The source of this asymmetry is not that computers are not dynamical systems, it is that there are fundamental limits to what humans can know about dynamical systems. In an important sense, in order to understand the scope of Turing Machines in the natural sciences it is important to develop a deep understanding of Chaos theory.

On the other hand, Turing's fixed point theorem demonstrates that the scientific method is fundamentally necessary. The reason why experiments are essential in the natural sciences is that even if all of science concerned deterministic systems whose evolution was defined by computable functions, Turing's fixed point theorem shows that the behaviour of many of these systems would escape the analytical process of the most talented team of mathematicians independently of the computational resources at their disposal.

Finally, I would like to note that Universal Turing machines are strictly more powerful than dynamical systems(which are equivalent to Turing Machines) because they possess unbounded memory as well as a combinatorial language that is sufficiently expressive to simulate any dynamical system. The subject of UTMs will be addressed in a future article.

References

[1] Church, Alonzo (1936). "An Unsolvable Problem of Elementary Number Theory". American Journal of Mathematics. 58 (2): 345–363. doi:10.2307/2371045. JSTOR 2371045.

[2] Alan Turing, On Computable Numbers, with an Application to the Entscheidungsproblem. A Correction, Proceedings of the London Mathematical Society, Series 2, Volume 43 (1938), pp 544–546

[3] Steven H. Strogatz. Nonlinear Dynamics and Chaos. CRC Press; 2nd edition. 2015.