THE PLANCK-CHURCH-TURING THESIS

Aidan Rocke
aidanrocke@gmail.com

May 3rd, 2021

Contents

1 The fundamental theorem of dimensional analysis

Motivation

If we know the appropriate physical physical units for the observable Ω then we have made significant progress in understanding its behaviour. The Buckingham-Pi theorem tells us how many physical units we need and what we can do with free parameters if we happen to have more physical measurements than what is necessary to model the behaviour of Ω.

The theorem and its proof

Let's suppose we have succeeded in identifying an equation that describes the evolution of Ω as a function of N physical units. Then we have:

$$\exists \alpha_i \in \mathbb{Z}, \Omega = \prod_{i=1}^{N} \omega_i^{\alpha_i} \tag{1}$$

where ω_i are our physical units(ex. Joules).

Now, we also know that each unit of Ω may be expressed in terms of the fundamental units $U = \{u_i\}_{i=1}^{k}$ so we have:

$$\exists \beta_i \in \mathbb{Z}, \Omega = \prod_{i=1}^{k} u_i^{\beta_i} \tag{2}$$

and for each physical unit,

$$\exists \lambda_{i,j} \in \mathbb{Z}, \omega_j^{\alpha_j} = \prod_{i=1}^{k} u_i^{\lambda_{i,j} \cdot \alpha_j} \tag{3}$$

In fact, we can model any complete system of scientific units as an Abelian group. Digression aside, this allows a representation in terms of the system of equations:

$$\sum_{j=1}^{N} \lambda_{i,j} \cdot \alpha_j = \beta_i \tag{4}$$

so we have

$$\Lambda \cdot \vec{\alpha} = \vec{\beta} \tag{5}$$

Now, in order to translate between different civilisations which might have different metrology institutes, it makes sense to model the problem using dimensionless parameters. This amounts to finding $\vec{\Delta\alpha}$ such that:

$$\Lambda \cdot (\vec{\alpha} + \vec{\Delta\alpha}) = \Lambda \cdot \vec{\alpha} = \vec{\beta} \tag{6}$$

where the set of all possible $\vec{\Delta\alpha}$ defines the null-space of Λ.

From the rank-nullity theorem we may deduce that:

$$\dim(\text{Null}(\Lambda)) = N - k \tag{7}$$

is the number of dimensionless parameters, constructed from the N original variables, with which we may describe our physical system.

A useful intuition

The reader might wonder why all units have integer-order dimensions. The reason for this is that the most fundamental interactions in physics are well-approximated by integer-order partial differential equations.

2 Proof-Program equivalence in the natural sciences

Introduction

There is a general impression in the computational biology community that theories of computation are both difficult to understand and useless. I believe this situation is partly due to gaps in the general education of computational biologists. This may be filled by motivating theoretical neuroscience from a biological perspective [1] as well as reading a book on the history of computation where each advance is carefully justified.

However, for those computational biologists that would like a short introduction I prepared this article which focuses on insights and intuitions. Starting with function composition, which all scientists are familiar with, I show that proof-program equivalence is a natural consequence. I then show that there is a direct correspondence between this functional approach to computation, from the perspective of a mathematician, and Turing Machines, developed from the perspective of a stored-program computer.

A natural definition of computation

If we define an approximate bijection between the sequence of mathematical operations $f = \{f_i\}_{i=1}^{n}$ of a function f with $Dom(f) = X$ and $Im(f) = Y$, such that for any $x_0 \in X$:

$$x_1 = f_1 \circ x_0 \tag{8}$$

$$x_2 = f_2 \circ x_1 \tag{9}$$

$$f \circ x_0 = f_n \circ x_{n-1} \tag{10}$$

and the sequence of mathematical operations $g = \{g_i\}_{i=1}^{n}$ on a system state $z_0 \in Z$, carried out by an engineered system:

$$z_1 = g_1 \circ z_0 \tag{11}$$

$$g \circ z_0 = g_n \circ z_{n-1} \tag{12}$$

then we say that x_n is an encoding of z_n, and g evaluated at z_0 computes the operation f at x_0.

We note that if the physical operation of the program halts at step n, then the program returns its state at that instant:

$$z_n = g_n \circ z_{n-1} \tag{13}$$

and in this setting, a finite loop is a sequence of functions $\{f_i\}_{i=1}^{n}$ where $n < \infty$ and:

$$\forall i, j \in [1, n], f_i \equiv f_j \tag{14}$$

At this point we may make a couple useful observations:

1. There are no infinite loops in a physical system as every physical computer runs on a finite amount of energy.

2. Biological systems address energy constraints via a complex combination of heuristics that are used for estimating the appropriate stopping time. In the real world, energy is a more important resource than time which is at most a proxy measure for energy. In any case, stopping criteria are an approximate solution to the halting problem in the real world [3].

Type theory and proof-program equivalence

Now, we note that the physical operation g_i and g_{i+1} compose with each other only if they share the same units so all physical equations define a type restriction:

$$\text{Type}(\text{Im}(g_i)) \equiv \text{Type}(\text{Im}(g_{i+1})) \tag{15}$$

so $g = \{g_i\}_{i=1}^n$ defines a path through a space of [scientific units](https://paulispace.com/post/2021/04/28/buckingham-pi.html) which may be modelled as a particular topological space [2].

For concreteness, let's suppose we would like to perform a back-of-the-envelope calculation to verify the proposition that the total mass of ants is approximately equal to the mass of humans:

$$\text{P} := M_{\text{ants}} \approx M_{\text{humans}} \tag{16}$$

where approximate equality is defined as a difference of less than one order of magnitude:

$$\frac{1}{10} \leq \frac{M_{\text{ants}}}{M_{\text{humans}}} \leq 10 \tag{17}$$

To perform a verification, i.e. the proof, we need a certain number of observables that are measurable(i.e. our assumptions).

$$\text{Ant axioms} := \text{Number of ants, Average mass of an ant} \tag{18}$$

$$\text{Homo sapiens axioms} := \text{Number of humans, Average mass of a human} \tag{19}$$

where we have distinct types for number(Integers) and mass(kilograms). It follows that we may define the input:

$$z := (N_{\text{ants}}, N_{\text{humans}}, m_{\text{ant}}, m_{\text{human}}) \tag{20}$$

$$g = g_2 \circ g_1 \tag{21}$$

$$g_1 \circ z = \frac{N_{\text{ants}} \cdot m_{\text{ant}}}{N_{\text{humans}} \cdot m_{\text{human}}} \tag{22}$$

and using Boolean arithmetic we have:

$$g_2 \circ g_1 = \text{Bool} \circ (g_1 \circ z \geq \frac{1}{10}) + \text{Bool} \circ (g_1 \circ z \leq 10) \tag{23}$$

and if we carry out the calculation, we find:

$$g \circ z = 1 \tag{24}$$

So our constructive proof defines a program. It follows that within the realm of scientific computation(a subset of constructive mathematics) the notions of proof and program are equivalent.

Turing machines for simulating physical systems

The functional perspective that has been introduced so far has been formalised as the Typed Lambda calculus which allows us to reason about the semantics of computation. This perspective is essential because computations are not observer independent. On the other hand, Turing Machines are an equally powerful model of computation that emerge naturally from the perspective of a stored-program computer. They offer a complementary perspective as they allow us to reason about the mechanics of computation from the perspective of a computer. In particular, they allow us to analyse the computational complexity of algorithms. Moreover, a direct correspondence between the two perspectives is relatively simple to establish.

In order to simulate a finite sequence of mathematical operations $\{f_i\}_{i=1}^n$ applied to $x_0 \in X$ on a realistic computing device, let's assume each state transition $x_n = f_n \circ x_{n-1}$ has a binary encoding, each f_i has a binary encoding and each state $x_i \in X$ has a binary encoding. It follows that we may refer to both functions and their variables $x \in X$ as mathematical symbols.

As all physically realisable computers have a finite amount of memory and at any instant the sequence of operations only depends on the state x_{n-1} and the function f_n being applied to it, we may model this process using a length of tape with countably many slots. We may imagine that each state transition is encoded by moving to the right on a finite length of tape and printing a sequence of ones and zeros with blank symbols($\emptyset$) between adjacent encodings of mathematical symbols.

Likewise, in order to define the next state-transition $f_{n+1} \circ x_n$, the computer must go back by moving to the left in order to replace the previous digits(our symbols) with new ones so the current sequence of symbols is in agreement with the binary encoding of $f_{n+1} \circ x_n$. (This corresponds to what computer scientists call garbage collection, an automatic form of memory management.)

It follows that we generally need a finite set of mathematical symbols which is a subset of the set of binary sequences with the empty symbol($\emptyset$):

$$\Sigma \subset \{0, 1\}^* \cup \emptyset \tag{25}$$

a set of states which in our case is a subset of the set of binary sequences as each state-transition $f_n \circ x_{n-1}$ is identifiable with a distinct state x_n:

$$X \subset \{0, 1\}^* \tag{26}$$

an initial state:

$$x_0 \in X \tag{27}$$

as well as final states i.e. termination criteria for the program:

$$\bar{X} \subset X \tag{28}$$

and our sequence of functions may be replaced by the mechanically-operated transition function:

$$\delta : (X \setminus \bar{X}) \times \Sigma \rightarrow X \times \Sigma \times \{\text{Left}, \text{Right}\} \tag{29}$$

where δ is generally a partial function due to the fact that the Halting problem is undecidable. Now, we may make a couple useful observations.

First, given a particular discrete-time dynamical system there exists a Turing Machine which may be used to simulate that system. Second, there are countably many Turing Machines. This means that if we can simulate $n-1$ discrete-time dynamical systems with $n-1$ distinct Turing Machines then we may identify the nth Turing Machine with a computer which may be used to simulate any of the $n-1$ dynamical systems. By induction, there exists a Turing Machine which may be used to simulate all the other Turing Machines.

We may call such a Turing Machine the Universal Turing Machine and we may identify it with the Universe itself which is actively being used to simulate every physical process we observe [6]. In the Universe, garbage collection happens in real time via competition for finite resources.

Discussion

Some scientists appear to struggle with the notion of an unbounded length of tape. But, I'd like to point out that this is not so different from an arbitrarily large envelope for back-of-the-envelope calculations. More importantly, a good understanding of the Church-Turing thesis is essential for all natural scientists because it defines the epistemic limits of the scientific method itself.

All physical systems at present are modelled within a mathematical framework that is consistent with Peano Arithmetic(PA) for combinatorics and algorithm design, first order logic for mathematical reasoning and proof verification, and a theory of types that is implicitly necessary for physical equations to make sense. This means that human scientists and the models they define operate within the confines of the Church-Turing thesis.

On the other hand, if physicists reasoned within a mathematical framework that was inconsistent with PA then scientific induction would no longer be possible. So they might develop theories but they would not be testable. It follows that if computational neuroscientists build a testable model of the brain, this model must be within the Turing limit. In a very precise sense the limits of our language define the limits of our world, but the language of mathematics is also unmatched in its expressive power as observed by Wigner [4].

In order to transcend these epistemic limits we may need to build more powerful computers using principles of black-hole engineering [5]. In the not-too-distant future, this will allow scientists to use scientific methods that are beyond the Turing-limit and therefore strictly more powerful than the ones we currently use.

References

1. Hessameddin Akhlaghpour. An RNA-Based Theory of Natural Universal Computation. Arxiv. 2020.

2. The Univalent Foundations Program. Homotopy Type Theory: Univalent Foundations of Mathematics. Princeton University. 2013.

3. Thomas S. Ferguson. Who Solved the Secretary Problem? Statistical Science, Vol. 4, No. 3 (Aug., 1989), pp. 282-289

4. Eugene Wigner. The Unreasonable Effectiveness of Mathematics in the Natural Sciences. 1960.

5. Hajnal Andréka et al. Closed Timelike Curves in Relativistic Computation. 2011.

6. Aidan Rocke (https://cstheory.stackexchange.com/users/47594/aidan-rocke), Understanding the Physical Church-Turing thesis and its implications, URL (version: 2021-02-22): https://cstheory.stackexchange.com/q/48450

3 Turing's fixed-point theorem

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

Introduction

Given the previous article on the correspondence between Turing machines and dynamical systems, I'd like to address the remaining objection among members of the computational biology community that dynamical systems with a finite energy budget always halt unlike Turing machines.

A fixed-point formulation of Turing's analysis of the halting problem demonstrates that the source of this asymmetry is not that dynamical systems may not be simulated by computers, it is that there are fundamental limits to what Turing machines(and therefore humans) can know about dynamical systems. Moreover, this formulation has another interesting feature as it clearly shows that that there is an intrinsic directionality in the information processing behaviour of Turing Machines.

Turing's theorem

Given f from the space of computable functions F and the metrisable space X, if we define the dynamical system:

$$\forall x_n \in X, x_{n+1} = f \circ x_n \tag{30}$$

then the existence of a general method H for deciding whether $\Lambda \subset X$ contains all attractors of f implies the existence of the fixed-point:

$$f \circ f^* = f^* \tag{31}$$

where $f^* \in F$.

Unfortunately, it may be shown that there is no such general method H which is guaranteed to halt within a finite number of operations.

Proof

Let's suppose we have a dynamical system given by:

$$\forall x_n \in X, x_{n+1} = f \circ x_n \tag{32}$$

where X is assumed to be a metrisable space and the nth term $x_n \in X$ is given by the iterated composition of functions:

$$x_n = f^n \circ x_0 \tag{33}$$

$$\forall n \in \mathbb{N}, f^{n+1} = f \circ f^n \tag{34}$$

then this sequence of operations defines a finite loop provided that $n < \infty$.

Now, if the space of attractors $\Lambda \subset X$ is given by:

$$\Lambda = \{x_0 \in X : \forall \epsilon > 0 \exists N \in \mathbb{N} \forall n \geq N, f^n \circ x_0 - f^{n+1} \circ x_0 < \epsilon\} \tag{35}$$

we may distinguish finite loops from infinite loops if there exists a general stopping criterion H:

$$H \circ f = \text{Bool} \circ \{\forall \epsilon > 0 \forall x_0 \in \Lambda \exists n \in \mathbb{N}, f^\infty \circ x_0 - f^n \circ x_0 < \epsilon\} \tag{36}$$

Assuming the existence of H, the question of whether Λ contains all the attractors of f is a decidable problem. Moreover, given H we may define:

$$f^* = \lim_{n\to\infty} f^n \tag{37}$$

which results in the fixed-point:

$$f \circ f^* = f^* \tag{38}$$

$$f^* : \Lambda \to \Lambda \tag{39}$$

where $f, f^* \in F$.

However, H requires an infinite number of function calls in order to specify fixed points so the application of H results in a paradox. From this we may infer that there is no general halting Oracle which may be used to determine whether Λ contains all the attractors of f.

By showing that such a function H does not exist, Turing showed that Turing Machines are constrained by a principle of limited omniscience. Turing's fixed-point theorem essentially points to the necessity of computer simulations in order to predict the future behaviour of a dynamical system whereas the Butterfly Effect places limits on what can be known from these simulations due to high sensitivity to the initial conditions.

Discussion

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 add that Universal Turing machines are strictly more powerful than dynamical systems(which may be simulated by Turing Machines) because they possess unbounded memory as well as a combinatorial language that is sufficiently expressive to simulate any dynamical system.

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.

4 The Planck-Church-Turing thesis

Motivation

In order to motivate the Planck scale, I think it helps to analyse the Generalised Uncertainty Principle where it naturally arises. What is even more interesting is the natural correspondence that emerges between civilisations incapable of Planck-scale engineering and civilisations within the Turing limit.

As Planck defined the physical limits of computer engineering, I think the Physical Church-Turing thesis is more appropriately named the Planck-Church-Turing thesis.

Plausible arguments for a Generalised Uncertainty Principle

Given that a photon has energy $h\nu$, and therefore an effective mass of:

$$M = \frac{h\nu}{c^2} = \frac{h}{c\lambda} \tag{40}$$

we should observe gravitational effects on this particle. This gravitational force should accelerate the particles, making its uncertain position even more uncertain.

Now, using Newtonian mechanics we may estimate the variation in acceleration and position due to gravity to be roughly:

$$\Delta a_g \approx \frac{GM}{r^2} = \frac{G \cdot (h/\lambda c)}{r^2} \tag{41}$$

$$\Delta x_g \approx \Delta a_g \cdot t^2 \approx G \cdot (h/\lambda c) \cdot \frac{t^2}{r^2} \tag{42}$$

where r and t denote the average distance and time of interaction.

The only characteristic velocity of the system is the photon velocity c, so we have:

$$\frac{r}{t} \approx c \tag{43}$$

so we obtain the following uncertainty due to gravity:

$$\Delta x_g \approx \frac{Gh}{\lambda c^3} \approx \frac{G\hbar/c^3}{\lambda} = \frac{l_p^2}{\lambda} \tag{44}$$

where l_p is the Planck length.

Given that $\Delta x \approx \frac{\hbar}{\Delta p}$ due to the Heisenberg Uncertainty Principle, we may derive the GUP by summing the uncertainties:

$$\Delta x \approx \frac{\hbar}{\Delta p} + l_p^2 \cdot \frac{\Delta p}{\hbar} \tag{45}$$

If we set $u = \frac{\hbar}{\Delta p}$, we find:

$$\delta(u) = u + \frac{l_p^2}{u} \tag{46}$$

$$\frac{d\delta(u)}{du} = 1 - \frac{l_p^2}{u^2} = 0 \Rightarrow u = l_p \tag{47}$$

and as a result, we have:

$$\Delta x \geq 2 \cdot l_p \tag{48}$$

Discussion

Planck first noted in 1899 the existence of a system of units based on the three fundamental constants:

$$G = 6.67 \cdot 10^{-11} m^3 \cdot kg^{-1} \cdot s^{-2} \tag{49}$$

$$c = 3.00 \cdot 10^8 \cdot m \cdot s^{-1} \tag{50}$$

$$h = 6.60 \cdot 10^{-34} kg \cdot m^2 \cdot s^{-1} \tag{51}$$

In particular, Planck found that these constants are dimensionally independent in the sense that no combination is dimensionless and length, time and mass may be constructed from them. In fact, the reader may check that this is equivalent to finding non-trivial solutions to the linear system:

$$3x + y + 2z = 0 \tag{52}$$

$$z - x = 0 \tag{53}$$

$$2x + y + z = 0 \tag{54}$$

where $x, y, z \in \mathbb{Z}$. However, the only solution is $(x, y, z) = (0, 0, 0)$.

Meanwhile, the energy associated with the Planck mass is given by:

$$E_p = M_p \cdot c^2 = 1.2 \cdot 10^{19} \text{GeV} \tag{55}$$

It's worth noting that observational analysis of Planck scale physics is highly problematic since modern high energy particle experiments are on the order of 10^3 GeV, and even the most high-energy cosmic rays detected are on the order of $\sim 10^{12}$ GeV which is far below the Planck scale. In fact, the Planck-scale effectively describes both the experimental and theoretical limits of the Standard Model and therefore the physical limits of computer engineering for human civilisations operating within the Standard Model.

Finally, given that a good understanding of physics at the Planck-scale would potentially allow us to engineer black holes capable of computing any limit-computable function within a finite amount of time there is a correspondence between civilisations within the Turing limit and civilisations within the Planck limit i.e. civilisations that have not yet mastered the principles of Planck-scale engineering.

References

1. RJ Adler. Six easy roads to the Planck scale. 2010.

2. B. Carr, J. Mureikab and P. Nicolini. Sub-Planckian black holes and the Generalized Uncertainty Principle. 2015.

3. RJ Adler, P. Chen, and D. Santiago. The Generalized Uncertainty Principle and Black Hole Remnants. 2001.

4. Domingos S.L. Soares. Newtonian gravitational deflection of light revisited. 2009.

5 Computations are observer-dependent

If the Planck-Church-Turing thesis is valid [1,2], the equivalence of work and computation would immediately follow:

$$\text{work} \Rightarrow \text{computation} \tag{56}$$

$$\text{computation} \Rightarrow \text{work} \tag{57}$$

However, this is not something we may take for granted as it has non-trivial implications. It would almost immediately follow that all physical theories are observer dependent.

Computations are observer-dependent in general as a theory of computation requires that the syntax and semantics of computations be expressed within a formal language. In fact, the epistemic limits of that formal system define a world model and so all computations are fundamentally Bayesian. Moreover, computations are the expression of mathematically precise relations between publicly available data so computations are meaningful only if they are simultaneously meaningful for a population of observers.

This means that a shared language must emerge through a population of biological organisms and this language must reach a degree of sophistication by some process of linguistic morphogenesis that is sufficient for the analysis of mathematical expressions. In particular, observations are both *mathematical relations* and *computations* because these are necessarily carried out by information-processing devices. But, what is the exact nature of these mathematical relations?

We would be wise to go back to Poincaré who noted that the scientist discovers the relations between things and not the ultimate nature of things themselves. In this context, the most fundamental physical observations are expressions of fundamental relations between humans and their environment. It follows that the general question of what is computable is a fundamental problem in epistemology and possibly the most important problem.

This will certainly require a deeper investigation of the three-way correspondence between Math, Physics and Computation or what is known as the Planck-Church-Turing thesis. I believe that an investigation into the foundations of mathematics which unifies math, physics and computation will require a deeper understanding of biological evolution and cosmological natural selection.

One approach to both cosmological natural selection and the foundations of math, physics and computation would be to analyse models of computation that are possible with black hole computers. Related work has also been done on the holographic principle.

References

1. Pauli Space. The Planck-Church-Turing thesis. 2021.

2. Aidan Rocke (https://cstheory.stackexchange.com/users/47594/aidan-rocke), Understanding the Physical Church-Turing thesis and its implications, URL (version: 2021-02-22): https://cstheory.stackexchange.com/q/48450

2. Hajnal Andréka et al. Closed Timelike Curves in Relativistic Computation. 2011.

3. Bertrand Russell. Human Knowledge: Its Scope and Limits. Routledge. 2009.

4. John A. Wheeler, 1990, "Information, physics, quantum: The search for links" in W. Zurek (ed.) Complexity, Entropy, and the Physics of Information. Redwood City, CA: Addison-Wesley.

5. Jeffrey M Shainline. Does cosmological evolution select for technology? Institute of Physics. 2020.

6. Poincaré. La Science et l'hypothèse. Champs sciences. 2014.

7. Chomsky, Noam (1956). "Three models for the description of language" (PDF). IRE Transactions on Information Theory (2): 113–124.