The Man in a Tailcoat

Tommaso Bolognesi (t.bolognesi@isti.cnr.it)

March 1, 2017

As a practical matter we often end up describing what systems do in terms of purpose when this seems to us simpler than describing it in terms of mechanisms - Stephen Wolfram

Prologue

Yesterday around midnight I had an interesting conversation with an elegant man. With a top hat and a curious glass walking cane, he was standing on the parapet of a bridge, looking down at the dark river below his feet. [1]

"What are your doing up there, Sir?", I politely asked. No answer. "Are you going to give up keeping your entropy low and far from thermodynamic equilibrium with the environment? Do you feel that your life has no purpose?" He turned his head and stared at me with a mix of surprise and irritation. "Well", I continued, "that's a bad idea. Purposes are overestimated concepts and an illusory business, both in life and elsewhere. Would you let me expand a little?"

I took his protracted silence as a consent, and proceeded.

Mechanisms and goals

Garey and Johnson [3] define a *problem* as:

> "a general question to be answered, usually possessing several *parameters*, or free variables, whose values are left unspecified. A problem is described by giving: (1) a general description of all its parameters, and (2) a statement of what properties the answer, or *solution*, is required to satisfy. An *instance* of a problem is obtained by specifying particular values for all the problem parameters".

Garey and Johnson are computer scientists. From their viewpoint (which may appear restrictive but is indeed fully general, as I'll clarify shortly) the solution of a problem is provided by an *algorithm*, which defines a step-by-step procedure, or computation, or process, or *mechanism*, for deriving from the problem instance (the input, or initial state) something (an output, or final state) that satisfies those required properties.

Terms like *purpose* and *goal*, that I use interchangeably, immediately fit into this picture: the *goal* of the *mechanism* is to satisfy the *properties* mentioned in point (2): you specify a goal by specifying those properties.

Here's an example of a mechanism m whose input is a tuple *in* of numbers, and whose details you don't need to grasp:

```
m[in_] :=  in //.
           {x___, a_?NumericQ, b_?NumericQ, y___} :> {x, b, a, y}
           /; b < a
```

And here is m's goal g whose details might look more familiar (for simplicity I assume the numbers of tuple *in* to be all different):

$$g(in, out) =_{def} (set(in) = set(out)) \wedge (\forall j \in [1, n-1].out(j) < out(j+1)) \quad (1)$$

[1] 'L'uomo in Frack' (The Man in a Tailcoat), also known as 'Vecchio Frack', is a poignant song by the italian singer and composer Domenico Modugno (1928-1994). Written in 1955, it describes the last hours of a mysterious character and is inspired by the true story of Raimondo Lanza di Trabia.

Figure 1: 'L'uomo in Frack' (painting by Dina Mosca)

Mechanism m is written in the *Mathematica* programming language, and operates by swapping a pair of elements a and b whenever a precedes b in the tuple and $b < a$, until no such pair can be found. It is a rather naive sorting algorithm.

Goal g is written in first-order predicate calculus: it establishes that tuples *in* and *out* have the same elements and that the elements of *out* are sorted.

The equations that summarise Garey and Johnson's quote are then:

$$m(in) = out \tag{2}$$

$$g(in, out) = True \tag{3}$$

Both m and g define an input-output relation R. Mechanism m does it in a constructive way, 'by addition': given an *in*, it creates an associated *out* and adds pair (in, out) to R. Goal g does it in a declarative way and, in a sense, 'by subtraction': it puts all potential pairs in R, and then removes those that do not verify the requirement. Note that both construction and verification involve a computation, and in both cases the computation may diverge, since it is undecidable whether a generic m terminates on a generic *in*, and it is undecidable whether a generic predicate $g(in, out)$ is provable by the rules of first-order predicate calculus.

In principle, then, code m is as good as logic formula g for characterising an input-output relation. In practice, however, as the mechanisms become more and more complicated, a compact logic formula (whenever available!) is much more preferable for concisely characterising the process at hand, and for referring to it in human-to-human communication.

(In the darkness it is hard to decipher my interlocutor's expression. Puzzled?)

I guess you are tempted to criticise my approach as being too restrictive, since: (i) it illustrates *artificial* processes, like computer programs, but ignores *natural* processes; (ii) it focuses on mechanisms of a computational type, leaving out all the others. Fine! This gives me the opportunity to clarify why the computer scientist's viewpoint about mechanisms and goals is fully general and universal. That's because the whole physical world is itself algorithmic and carries out a gigantic computation, a relentless information processing activity that started with the Big Bang.

(At this point, The Man in a Tailcoat gave a discreet cough.)

Thus, the correct way to look at a collision between two subatomic particles, a lightning, a biochemical reaction, the appearance of a new species, is in terms of a computing process. In this respect, the distinction between *natural* mechanisms, like these, and *artificial* mechanisms, like the computation of a sorting algorithm, is blurred. All we have is mechanisms - interacting, computational mechanisms all over the place!

"And goals? Do all mechanisms have a goal?" you may ask. (This seems to be a question of vital importance for The Man in a Tailcoat, tonight.)

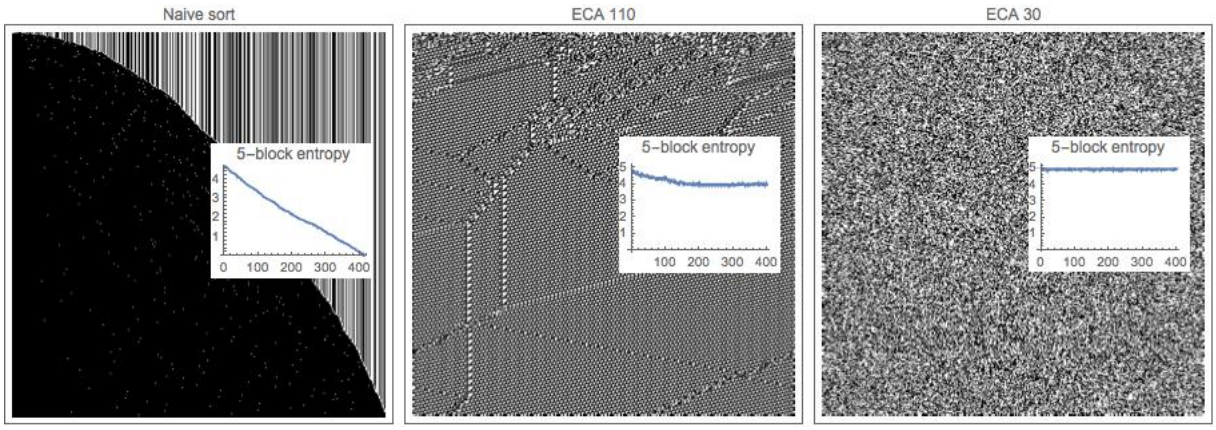

Figure 2: 5-block entropies for the naive sorting algorithm, ECA 110 and ECA 30 (sampled).

For artificial systems, finding the correct (m, g) mechanism-goal pairing is usually trivial, since g is defined a priori by humans and the matching process m is designed by humans too, a posteriori and in a short time. This is the engineer's business.

But try to figure out g a posteriori, that is, by just looking at the program code! This is what happens with natural mechanisms. In nature no goal seems to be established a priori, by some Grand Architect, and we know today that the design of the various mechanisms, at least for the complex processes associated with life, is carried out by blind darwinian evolution, taking evolutionary times up to hundred-million years. Still the question remains: is it possible to find a concise formulation of a goal $g(in, out)$ for any natural mechanism m we observe, in the computation-oriented setting discussed above? This is the scientist's business, and for doing it our reasoning in terms of (in, out) pairs needs to be revised.

Compact goals, compressible mechanisms, decreasing entropies

Many of the artificial mechanisms we build, e.g. sorting algorithms, are adequately characterised by their input-output behaviour, and by the compact $g(in, out)$ logical formulae conceived for defining their goals. When dealing with natural mechanisms this approach may become less effective, since the precise output of these processes is often hard to define. Rather, the mechanisms tend to run forever, so the $m(in) = out$ scenario must be revised in favour of a potentially endless $m(in)$ computation. For example, what is the final state of a planet orbiting around a star? On one hand, each individual step of the process is relevant and potentially eligible as an *intermediate* output of the computation, and we might ask what is the goal of the process *up to that point.* [2] On the other hand, when the output becomes a moving target, a reasonable alternative to conceiving m's goal as a $g(in, out)$ property is to focus on the internal features of m in all its span, and replace the requirement of a clear and concise logical formulation of the (dynamic) input-output relation by alternative 'lightness' requirements.

For investigating alternative definitions of purposefulness, let us consider an example. In Figure 2 we compare three mechanisms: the already introduced naive sorting algorithm and two elementary cellular automata (ECA) – ECA 110 and ECA 30. For ease of comparison we sampled the computations: the diagram for the sorting algorithm shows only every 100th row, while for the ECA's the sampling period is 8. Furthermore, rows are bit tuples of length 400 in all three cases. ECA's do process bit tuples. For the sorting algorithm, intermediate configurations would be tuples of integers: $(n_1, ..., n_{401})$. We turn them into bit-tuples $(b_1, ..., b_{400})$ by comparing adjacent numbers: $b_i = 0$ (resp. 1) when $n_i > n_{i+1}$ (resp. $n_i < n_{i+1}$).

Which of these mechanisms would we happily regard as purposeful?

A concisely expressed goal for the sorting mechanism (Figure 2-left) was established a-priori by

[2] In the context of the computational universe conjecture, the usual answer to the question "What is the *goal* of the universal computation?" is: "To compute the universe's own evolution" [5]. At each step the goal of the computation has been to push the universe up to that point. Tautological as it may sound, the answer has a genuine message: there is no clear and concise way to describe the behaviour of the universe other than by replicating it step-by-step. Another way of saying this is that the computation is not compressible.

equation (1). Having turned numbers into bits, the goal component dealing with the comparisons of adjacent elements turns into the requirement for the output tuple to contain only 1's (black cells).

What about the computations of ECAs 110 and 30? As observed in [7] (p. 830), the first automaton *gives the impression* of being purposeful, the second does not. How can we substantiate these impressions?

When started from a random bit tuple, ECA 110 quickly develops digital particles, still perfectly visible with our periodic sampling (Figure 2-center), that appear to interact and carry out themselves a computation. Indeed this automaton is Turing universal [2] and the motions of its particles can be exploited for simulating any (purposeful) algorithm, including sorting. Furthermore, with ECA 110 one can still somehow preserve the $g(in, out)$-oriented view. If in is some input bit tuple where a particle is already formed, and t is a predefined number of computation steps, one can predict the form of the output row out, due to the rectilinear motion of the particle: roughly, out will consist of the periodic background with the particle positioned at a distance s from its initial position, defined by a $s = vt$ type of formula. A careful study of particle collisions would allow to extend the approach to more complex scenarios. And we could again express the input-output relation $g(in, out)$ *concisely*.

With ECA 30 (Figure 2-right) the noise-like behaviour seems to prevent the pursuit of any meaningful goal.

Both for sorting and (to some extent) for ECA 110 we can find concise logical formulations of input-output relations. And both computations can be compressed. In the original sorting algorithm a pair of numbers (a, b) is considered at each step, and possibly swapped: one can compress the computation by considering, say, two appropriate pairs at a time. The compressibility of the ECA 110 computation is apparent from the sampled diagram, where particle interactions are preserved. Compressibility also derives from the highly regular structure of the background.

On the other hand, we could not associate a goal to ECA 30, and its computation is not compressible: to know the n-th row requires to compute all the preceding rows, one after the other. This circumstance may suggest to see compressibility as a necessary and sufficient condition for purposefulness. Would you agree?

(The Man in a Tailcoat remained silent and stared off into the distance. No, he does not seem to agree. Why?)

Oh, you are right! This would be a badly circular definition. We want to define purposefulness in terms of compressibility, but compressibility means reducing the length of the computation *without affecting its functionality*, that is, while preserving its original purpose! How can we safely compress a computation without knowing its purpose? How would we tell apart redundant from essential bits? Put it differently, if a purposeful mechanism must be compressible by definition, I could compress it until it becomes incompressible, without affecting its purpose: the result would be an incompressible, purposeful mechanism. A contradiction!

We can conclude that *we cannot take compressibility as an indication of purposefulness*. Well, thank you, my friend, you had a good point there! (The Man in a Tailcoat, still standing on the parapet, does not react visibly to my gratifying remark.)

Let's now take a step forward (I mean... only metaforically!). There is another rigorously measurable quantity whose fluctuations might reveal the presence or absence of a goal: *entropy*.

The very peculiar entities that appear to manifest goals in this world – and intentions, consciousness, agency – have developed a great ability to maintain their entropy low, and lower than that of their environment. In a universe otherwise dominated by the 2nd principle of thermodynamics, this is indeed the mother of all goals – a tough job whose pursuit has been carved by darwinian evolution into the conscious and unconscious habits of living creatures, and one of the defining features of the emergent phenomenon of life. But just how early does this attitude emerge in the physical world? How much earlier than the appearance of the biosphere? What is the simplest mechanism, or system of open, interacting mechanisms in which localised entropy reduction may take place? (I extract a candle from my pocket, place it near the man's feet on the parapet, and light it up for illuminating the next figure.)

Let's consider the entropy fluctuations for the three processes of Figure 2 (inset diagrams). For the two mechanisms that we recognise as purposeful - the sorting algorithm and ECA 110 - entropy decreases; for the purposeless ECA 30, entropy remains stable to its maximum possible value 5.

The notion of entropy we have adopted here is Shannon entropy $-\sum_i p_i Log_2 p_i$, relative to bit-blocks of length 5. For each 400-bit row of the computations, the p_i's were obtained by counting the number of occurrences of the bit-tuples of length 5 - an arbitrarily chosen length. (Then, the 5-block entropy of a completely random bit row is 5 because the 32 distinct 5-blocks are equiprobable, with $p_i = 2^{-5}$.)

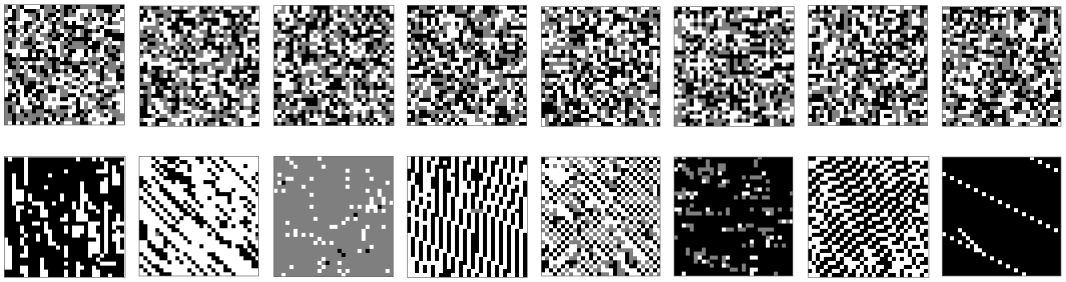

Figure 3: Upper: random typing on eight paper sheets, using a three-character (three-color) typing machine. Lower: results of 80,000-step computations of eight randomly chosen 2-dimensional, 3-state, 3-color Turing machines, each running with the corresponding sheet above as input. (From [1])

Expectedly, the decrease of entropy in the first two examples of Figure 2 corresponds to a growth of perceived order and regularity. In a computational universe consisting of all conceivable mechanisms – random algorithms operating on random inputs, or monkeys typing at random on computer keyboards – it is well known that the outputs follow the universal a priori probability, or Solomonoff-Levin algorithmic probability [4], with a more pronounced bias towards regularity (Figure 3 - lower) than in a truly random universe – monkeys typing on plain old typewriters (Figure 3 - upper). Thus, if the conceptual equation linking goals to local entropy reduction is correct, we could say that the existence of mechanisms with a goal in the computational universe does not come as a complete surprise...

(In fact, my interlocutor does not look surprised at all, at the moment. Bored?)

Ok, I grant you that this may sound too abstract. In particular, the eight patterns in the lower part of Figure 3 are the outputs of eight mechanisms that have proceeded in complete independence from one another for 80,000 steps. We could do better by letting mechanisms cooperate.

A closed physical system – one unable to interact with its environment – cannot hope to invert the natural tendency of its entropy to grow. An open system can. Our Earth, for example, is an open system that interacts with outer space. As observed in [6], the energy it receives from the sun during the day is roughly the same it returns to dark space at night, but, due to the large temperature gap between the sun and dark space, the incoming photons are individually more energetic, thus less numerous, than the outgoing, less energetic photons. Less incoming than outgoing photons implies less incoming than outgoing entropy. This is the way our planet manages to keep order – low entropy – in the biosphere.

Consider now a toy computational universe consisting of the seven ECA's $\{41, 48, 54, 98, 102, 158, 206\}$ acting independently from one another, like the Turing machines of Figure 3 - lower row. The entropy levels attained by these automata are shown in the left diagram of Figure 4. In the simulations we have run each ECA for 1 million steps, starting from a random 600-bit array, and have computed the 5-block entropies of every 1000-th step, obtaining a plot of 1000 entropy values for each automaton. In spite of the wider oscillations of ECA's 54 and 102, each ECA settles quickly to its own peculiar level, ranging between value 3.3 achieved by ECA 98 and a value a little below 5 achieved by ECA 102.

What happens if we let the ECA's cooperate? In this slightly less naive universe we have arranged the automata in a cycle: $41 \to 206 \to 54 \to 98 \to 158 \to 48 \to 102 \to 41$, so that the output of one is input to the next. Each automaton runs for 1000 steps, and a total of 1000 iterations are performed, as in the previous simulation.

The interesting effect of cooperation is to spread entropy values in a wider range. In particular: the values of ECA's 102 and 41 are not affected; those of ECA's 54 and 98 slightly increase; those of ECA's 48, 158 and 206 *decrease*. The entropy decrease of ECA 48 is remarkably close to one unit!

In traditional living organisms one can in principle establish an entropy threshold above which the organism is dead. Assume we decided: (i) to attribute an alive/dead status to the seven inhabitants of our toy universe, (ii) to do it just in terms of their entropies, and (ii) to set their life-to-death entropy threshold at value 3. Under these assumptions, we should conclude that none of the automata is alive in the scenario of independent systems, while one of them – ECA 48 – would come to life when they cooperate. You may certainly dismiss these bizarre assumptions as a blatant over-simplification of the multiple requirements behind the complex phenomenon of life. I would agree. This is meant to be just a proof of concept. But the concept it proves is interesting: the game of local entropy reduction, that

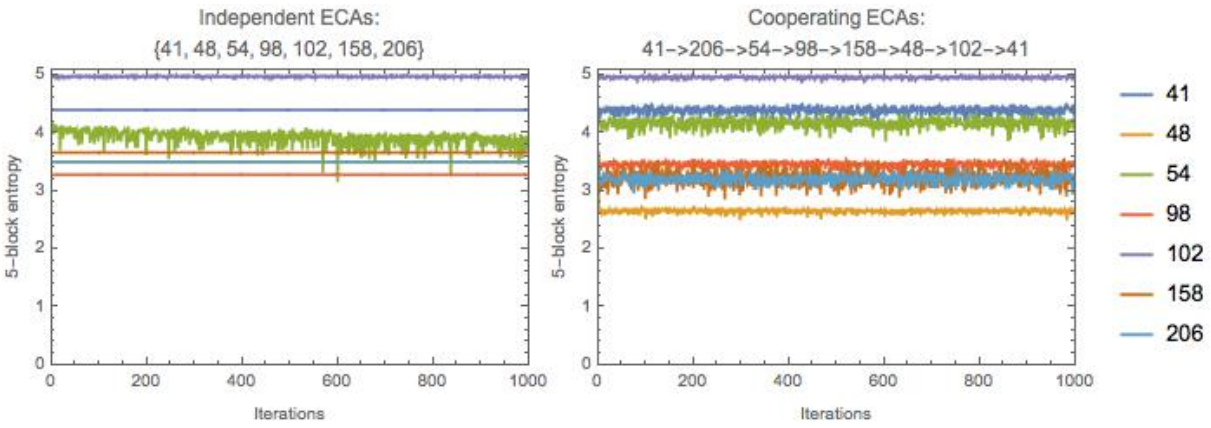

Figure 4: Entropies of seven ECAs acting independently (left) and in cooperation (right).

we recognise as a crucial ability of living organisms, may well be played also among formal, abstract algorithms that are normally considered as totally extraneous to the mechanisms of biology.

(The starry night must have reached its coldest peak. As dawn approaches, my silent interlocutor keeps staring at the river below us, as if dangerously attracted by the highly entropic flow of low-frequency photons that keep escaping the planet.)

Ok, my friend, you are right: I started our conversation by mentioning the illusory nature of goals and purposes, but have not yet justified that claim. Let's do it now.

Goals and emergence

Simply stated, my point is that mechanisms exist in nature as the fundamental building blocks of External Reality: they enjoy the most respected ontological status. Goals don't. Goals are only a convenient product of human knowledge, an epistemological device, a mental construction meant to offer practical representations of the mechanisms we observe *in the upper levels of the universal architecture of emergence*, those that we inhabit ourselves. Understanding those mechanisms in terms of compactly expressible goals has indeed given a formidable evolutionary advantage to our species, much as the development of human languages did, and this explains why goal-oriented reasoning is so widely spread and why we attribute so much importance to it.

I do insist that goals tend to be more easily formulated as we proceed upward in the hierarchy of emergence. Up in the highest levels, in the biosphere, the illusory nature of goals is well understood and accepted: they are just a *powerful narrative trick* for describing, a-posteriori, features of mechanisms that darwinian evolution developed without any a-priori blueprint. We are so much accustomed to goal-oriented story telling that we sometimes attribute intentions even to computers, cars, vacuum cleaners, when we describe their behaviours in anthropomorphic terms!

But what about the existence of goals as we proceed *downward* and look at very simple computational mechanisms? Recall the formulation of mechanisms and goals in terms of, respectively, an algorithm $m(in)$ and a logic formula $g(in, out)$, and the requirement that g be a *compact* logical predicate that can be more readily understood and communicated than the description of mechanism m (the tradeoff being, in general, that g defines the input-output relation in a non-constructive way). Now, let us again consider ECA 110. What made us think that ECA 110 has a goal is, ultimately, the emergence of particles. As already mentioned, at least in the case of a single particle we can conceive compact formulations of input-output relations – involving the initial and final position of the particle – essentially in terms of the laws of rectilinear motion, while completely ignoring the corresponding mechanism. The mechanism, of course, consists in the *interplay of the multitude of cells* that make up the automaton, all characterised by the boolean function $f : \{0,1\}^3 \to \{0,1\}$ that defines the next value b' of a cell in terms of the current value b of the cell itself and of the values a and c of its left and right neighbors: $b' = f(a, b, c)$. Thus, when we focus on particles, at the viewpoint of emergent Level 1, we can successfully distinguish between goal and mechanism, as in many biological processes. Can we do the same when moving our viewpoint down to Level 0? In other words, can we distinguish between a goal and a mechanism when looking at a *single step* of an *individual* ECA 110 cell?

The boolean function for ECA 110 cell behaviour is:

$$b' = f(a, b, c) =_{def} Xor[Or[a, b], And[a, b, c]] \tag{4}$$

You may suggest that equation (4) describes a *mechanism* since it is expressed in terms of more elementary boolean functions (*Xor*, *Or*, *And*) whose 'execution' can be carried out in a sequence of computational steps.

How about a corresponding *goal*? The only alternative representation that comes to mind – excluding an equivalent combination of different basic boolean functions – is a truth table, i.e. a rather lengthy exhaustive enumeration of input-output pairs of form $(a, b, c) \rightarrow b'$:

$$\begin{aligned} &\{(1,1,1) \rightarrow 0, (1,1,0) \rightarrow 1, (1,0,1) \rightarrow 1, (1,0,0) \rightarrow 0, \\ &(0,1,1) \rightarrow 1, (0,1,0) \rightarrow 1, (0,0,1) \rightarrow 1, (0,0,0) \rightarrow 0\}. \end{aligned} \tag{5}$$

You might then propose (5) to be the goal.

The problem I envisage in these stipulations is that (5) is not really an alternative formulation of (4) but only a direct, brute-force representation of its semantics.

To summarize: at Level 1 (particles), I can conceive totally independent descriptions for the goal – dealing with emergent particle dynamics – and the mechanism – dealing with the cooperative behaviour of a multitude of cells, each behaving according to function f. At level 0 (individual cell performing one step) I can't. A single step of a single cell turns out to be too simple a phenomenon for admitting a double characterisation in terms of a mechanism *and* a goal: at Level 0 the distinction between the two is blurred.

This leads us to the conclusion that goals prosper as levels of emergence start to pile up, even outside the realms of traditional biology.

Likewise, for talking about ECA entropy and its decrease, as a sign of purposefulness, we had to take the global picture: multiple cells, multiple steps. How could we detect entropy decrease, or even talk about the entropy, for a single cell doing a single step? And, for that matter, how could we talk about the thermodynamic entropy of a single particle in an empty volume of space? For entropy to be definable, we need a multitude of particles, and two levels; micro-states and macro-states with their observable macro-variables, right? Take Boltzmann's formula...

Epilogue

For reasons that are still obscure to me, pronouncing the famous physicist's name must have triggered a deadly flow of neural microstate changes in the head of the man, with visible consequences on the macro-variables characterising both his facial expression and his precarious equilibrium on the parapet. "Are you perhaps a theoretical physicist yourself?" I asked, while abruptly grabbing and pulling the lower end of his walking stick in an attempt to stop an unwholesome action. Smooth and slippery as it was, the crystal cane remained firm in my hand, not in his. The end of this story of local entropy growth - too sad to be reported here - is told by the final verses of the song 'The Man in the Tailcoat'.

References

[1] Tommaso Bolognesi. Spacetime computing: Towards algorithmic causal sets with special-relativistic properties. In Andrew Adamatzky, editor, *Advances in Unconventional Computing - Vol. 1: Theory*, volume 22 of *Emergence, Complexity and Computation*, chapter 12, pages 267–304. Springer, 2017.

[2] Matthew Cook. Universality in elementary cellular automata. *Complex Systems*, 15:1–40, 2004.

[3] M. R. Garey and D. S. Johnson. *Computers and Intractability - A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, 1979.

[4] M. Hutter, S. Legg, and P. M. B. Vitanyi. Algorithmic probability. *Scholarpedia*, 2(8):2572, 2007. revision 151509.

[5] Seth Lloyd. Universe as quantum computer. *Complexity*, 3(1):32–35, 1997.

[6] Roger Penrose. *Cycles of Time*. The Bodley Head - London, 2010.

[7] Stephen Wolfram. *A New Kind of Science*. Wolfram Media, Inc., 2002.