Emergentism and Computational Limits to Knowledge

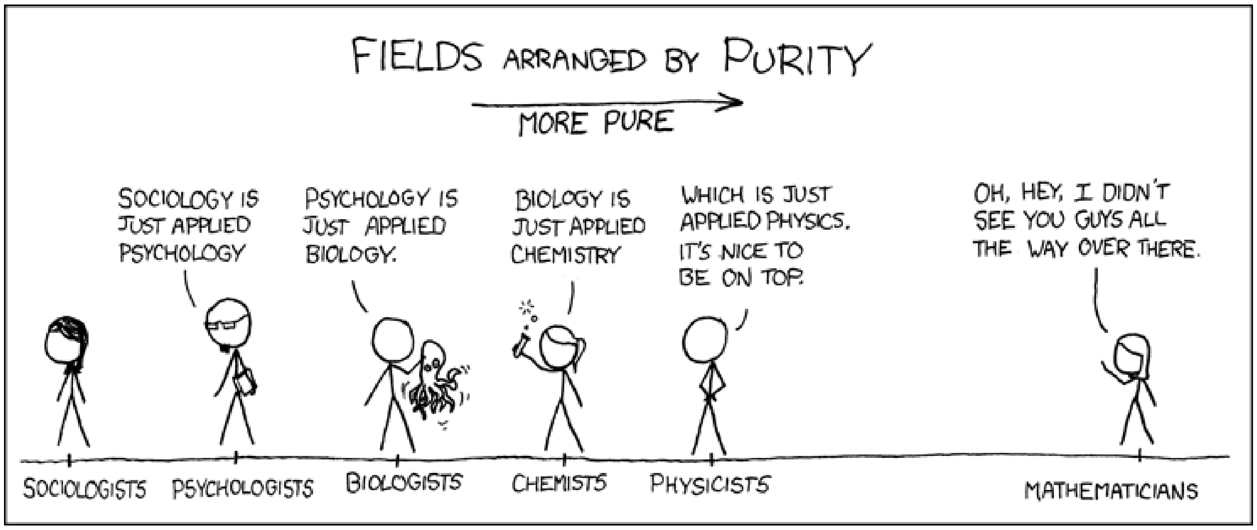
Anti-Climacus

November 15, 2023

In this note, I will argue that computational complexity theory has implications for the philosophy of science, specifically for ideas related to emergence and reductionism. In particular, I claim that if a property of a system is *computationally irreducible* in a certain sense, then it is arguably "emergent," and specifically "weakly computationally emergent."

What are emergentism and reductionism?

Reductionism is the dominant paradigm of modern science. Roughly speaking, it asserts that scientific knowledge can deduced from certain foundational rules and objects which give rise to all "higher-level" phenomena. This comic, which can be seen on the hallway doors of many physcists and mathematicians[1], explains reductionism by comparing various scientific fields[2].


FIELDS ARRANGED BY PURITY
MORE PURE
SOCIOLOGY IS JUST APPLIED PSYCHOLOGY
PSYCHOLOGY IS JUST APPLIED BIOLOGY.
BIOLOGY IS JUST APPLIED CHEMISTRY
WHICH IS JUST APPLIED PHYSICS. IT'S NICE TO BE ON TOP.
OH, HEY, I DIDN'T SEE YOU GUYS ALL THE WAY OVER THERE.
SOCIOLOGISTS
PSYCHOLOGISTS
BIOLOGISTS
CHEMISTS
PHYSICISTS
MATHEMATICIANS

Source: xkcd Comics.

The IEP defines reductionsim as such:

> *In what follows, the theory to be reduced will always be referred to as the target theory (T). The theory to which one is attempting to reduce the target theory will be known as the base theory (B).*

[1] My favorite rendition of this comic was by a pure mathematician, who went to the trouble of drawing an extra stick figure far behind the mathematician labeled "logician." You can guess the topic of their research.

[2] As a counterpoint, I endorse Paul Krugman's take on the matter - "*When I went to graduate school, I took international trade from Jagdish Bhagwati who explained to his class his personal theory of reincarnation which was that if you are a good economist, the virtuous economist, you are reborn as a physicist. But if you are an evil, wicked economist, you are reborn as a sociologist. By which he did not mean, by the way, that sociologists are bad people. What he meant is that the subject is even harder than economics to do.*" Source. The infamous Alan Sokal agrees.

There are three main ways in which reduction has been understood since the 1920s. These may generally be stated as follows:

Theory T reduces to theory B when all of the truths of T (including the laws) have been translated into the language of B.
Theory T reduces to theory B when all of the laws of T have been derived from those of B.
Theory T reduces to theory B when all of the observations explained by T are also explained by B.

Following the IEP, let's call these 3 positions "translationsim," "derivationism," and "explanationism." I am mainly interested in the third.

In my view, the best arguments for reductionism are that it has had great success in practice, and that it is parsimonious[3]. The argument for practical success is weaker since it is hard to guess what scientific advances we might have made in a world with a non-reductionist paradigm. Moreover, while reductionism is the official paradigm of science "on paper," it fails to match the day-to-day thinking and heuristics of scientists in practice. In mathematics, we have the old joke:

Mathematician: Mathematics used to be non-rigorous before the 20th century, but now we know that we are on sure footing because all of our vast and diverse results are reducible to the axioms of set theory.
Non-mathematician: What is set theory?
Mathematician: It is an elegant "grand unified theory" that explains all of modern mathematics.
Non-mathematician: So what are the axioms of this set theory then? Can you list them?
Mathematician: Oh, I have no idea what the axioms actually are. I never have to use them.

As for the argument from parsimony, here I think there is more room for debate. In particular, in this note I want to argue that the third property of reductionism defined by the IEP ("explanationism") is insufficient to "explain" properties of certain computational systems.

What counts as an explanation is the entire debate here; part of what I want to do is to elevate computational work from its second-class role in the existing discussions around this issue. We are used to thinking of the computations required to deduce a fact "merely calculations," but I will argue that this is untenable in light of the limits to computation.

What is emergentism then? Here I am less able to give a concrete definition, since much of this discussion depends on fuzzy and ill-defined concepts like "structure," "explanation," and so on. Rather than attempting to give a fully defensible definition of "emergence," I will simply define

[3] For example, Ethan Siegel argues in this essay that rejecting reductionism amounts of a "God of the Gaps" assumption in science.

my position as being against reductionism in certain specific instances, and then arrive at a property (*computational irreducibility*), that is arguably sufficiently "emergent." At the end of this note I will discuss certain definitions of emergence and see how they might apply to this argument.

Limits to Computation.

Computational complexity theory is a subfield of computer science and mathematics that studies the inherent limits to computation. We say that two computational problems A and B have the same complexity if they require the same computational resources (e.g. runtime, working memory) to solve. A gem of this theory is the classification of NP problems, which equates the computational complexity of literally thousands of computational problems that arise widely in science, engineering, and mathematics. NP problems are believed to be easy to verify, but hard to solve. For example, the following problem is in NP[4].

Generalized Sudoku: For some positive squared integer n (n = 9, 16, 25, 36….), and an n-by-n table with some entries filled in with numbers between 1 and n, fill in the rest of the entries so that each row, column, and each subtable of size square-root(n) by square-root(n) has each of the numbers {1, 2, …, n} exactly once.

Under the widely-believed Exponential Time Hypothesis (ETH), NP problems cannot be solved in anything less than exponential time. This requirement is so high, it makes the solution practically out of reach[5]. For moderately sized problem instances, the amount of time required to solve them is so massive that the heat death of the universe would occur before we could complete the computation.

Cellular Automata.

In this note, I want to focus on a different class of NP problems that correspond to cellular automata. Cellular automata (CA) are a class of computer programs (algorithms) that became popular in physics and mathematics research beginning in the 1980s. What makes these programs interesting are that they have two, seemingly incompatible properties:
(i) Locality: They change over time based on a short list of simple rules, each of which only uses local information (e.g. the neighboring "cells")
(ii) Global Structure: They can result in rich, complex global structures from a variety of initial conditions.

[4] Technically it is NP-complete. For the sake of accessibility to a broader audience, I will say "in NP" instead of "NP-hard" or "NP-complete" throughout this note; apologies to any computer scientists.

[5] Assuming certain statements about physics, and in particular the extended Church-Turing thesis. In particular, we'd have to rule out the possibility of different types of computation, based on physics we don't yet know. Quantum computation is one kind of computational model which is meaningfully different from the computers we use today. However, it should be noted that even quantum computers cannot solve NP-hard problems in reasonable time-scales. Assuming that Grover's algorithm is optimal, quantum computers still require superpolynomial time to solve NP-hard problems in general; this would not change the implications of this note.

Figure: A 2D cellular automaton defined by simple local interactions can result in intricate global structure. Source: Chinghang Lee.

Without getting into the weeds too much, there are a variety of prediction-type problems related to cellular automata that are known to be NP. See, for example, this article.

Therefore, let me offer a stylized example of an NP problem related to CA.

Cellular automaton state prediction: Given a cellular automaton with a certain set of rules, does there exist an initial state X which reaches state Y in at most T timesteps?

Note that this problem is completely straightforward if we don't care about computational resources, as we can just simulate the automaton on every possible initial state X for T timesteps, and check if one of them reaches Y. There are only finitely many states to start at, so this computation will halt in finite time.

Type A versus Type B knowledge.

Based on the discussion above, the answer to the cellular automaton state prediction problem is knowable in principle. In fact, the answer isn't even philosophically interesting, since the algorithm to learn the answer is just "try everything." This is characteristic of NP-hard problems; we believe that exhaustive search is the best algorithm for solving any such problem in general, such as Sudoku, Protein Folding, Traveling Salesman, and so on.

However, in reality we don't have infinite computational resources, and therefore cannot know the answer to the state prediction question for any cellular automaton with, e.g. 1,000,000 cells. Note that 1 million cells is not a big problem instance - it can be stored in a few megabytes! Nevertheless, running exhaustive search is completely intractable for even this moderately sized problem. Even if we converted the entire observable universe into a supercomputer, physical limits to computation such as the Bekenstein bound quickly limit the total amount of computations we might perform. Unless we can somehow harness computations occurring outside of our light cone, we are out of luck.

Therefore, we cannot know the answer to the cellular automaton state prediction question, assuming our current knowledge of physics and computation is correct[6]. More generally, we see that there are certain kinds of questions whose answer is knowable in principle (given enough computation) but is unknowable in practice (given our knowledge of physics).

This motivates the following definitions.

Type A: Knowable in this universe. These are inductive statements about our universe (e.g. the radius of Pluto) and formal propositions that are solvable in a physically possible computation. Example: "Does this cellular automaton on a 3x3 grid, starting on state A, ever reach state B?"

Type B: Knowable in some conceivable universe. Certain inductive and deductive statements fall into this category, and possible other kinds as well. In particular, this includes statements that require more computational resources to solve than our universe has.
Example (deductive): "Does this cellular automaton on a 10,000 x 10,000 grid, starting on state A, ever reach state B?"
Example (inductive): What was the date and location of the first instance of human-made fire?

Type C: Knowable in no conceivable universe. These statements are formal propositions that are undecidable. It seems that these statements should not be truth-apt, or falsifiable.

[6] I am also implicitly assuming that human brains cannot outperform computers on these computational problems. It's interesting to consider the opposite view, but it's hard to see why our brains would be particularly well-suited to predicting the states of automata. Much of the empirical evidence for the view that our brains are excellent reasoners really refers to the success of heuristics, which may work on the limited subset of problem-types that appear in our daily lives, but have no correctness guarantees in general. Moreover, the Bekenstein bound does apply to the region of space containing your brain just as much as it does to the region containing your laptop, but people who claim brains are not Turing Machines would probably reject the validity of applying the bound to brains.

Example: The Godel sentence arising from the proof of Godel's second incompleteness theorem.

If we didn't care about computational limits to knowledge, then Types A and B would coincide. A critic of my view might call the fact that we can know things about 3x3 cellular automata, but not 10,000x10,000 ones, "merely contingent." I agree! My point is precisely that, due to contingencies about our physical universe, certain deductive statements are simply impossible for us to know in general. Of course, we may get lucky and find that for a particular automaton of interest, our computation halts after only 500 steps. But if computational lower bounds are true, then this kind of luck provably runs out.

In addition to this practical reason to separate Types A and B, it also has the interesting consequence of demarcating *deductive* knowledge into two categories. While we are comfortable thinking of certain inductive facts as practically inaccessible (e.g. due to the Heisenberg uncertainty principle), it seems strange that deductive reasoning also has practical limitations. Indeed, the entire point of deductive reasoning is that it is free from the practical needs for laboratories, experimental apparatuses, and so on. Nevertheless, computational limits imply that even symbolic, abstract reasoning is ultimately a finite resource.

Computationaly irreducible states are emergent properties.
While the distinction between Type A and Type B knowledge applies to any NP-hard problems, let me recall the two special properties of cellular automata that I highlighted.
(i) Locality: They change over time based on a short list of simple rules, each of which only uses local information (e.g. the neighboring "cells")
(ii) Global Structure: They can result in rich, complex global structures from a variety of initial conditions.

Based on condition (i), cellular automata are reductionist by design; they are by definition reducible to individual cells, which only interact with their neighbors. As for (ii), any reductionist can (quite reasonably) argue that the emergence of global structure is simply a byproduct of the local interactions of cells over time.

However, when we consider the state prediction problem, we see that knowing the individual constituents of the system is not enough to "know" everything about the system. In particular, even the simple question of "will this ever reach state Y" is not knowable, at least according to the Type A definition we gave earlier.

One can object that the difficulty of the state prediction problem is in finding the correct initial state X, and once this is known then simulation is easy. In this view, the state X is what contains the cell-by-cell state information which characterizes the system, and to give a reductionist account of the system requires knowing the initial state.

To sharpen the discussion, we can formulate a version of the problem which avoids these difficulties.

State Transition Problem: Given a cellular automaton with initial state U, will it ever reach the state V?

Unlike the earlier problem, this one does not require that the terminal state is reached in some bounded amount of time[7]. It can still be solved in finite time, by simulating the automaton starting at state U. Either it reaches V, or returns to state U without reaching V. In the latter case we know the automaton is stuck in an infinite loop and can answer "no." Moreover, since there are only finitely many possible states of the automaton, this process takes finite time.

In this formulation, the problem is certainly still computationally hard, and so we run into the same issue of non-knowability (Type B knowledge) as before. However, we know both the full initial state and the rules of state evolution for the automaton. We arrive at the following conclusion:

There are certain systems for which, given (i) a full description of their initial state, and (ii) a full description of their dynamics (which are deterministic and involve only local interactions), we cannot (practically) know the answer to many simple questions about the systems. In particular, we cannot know whether they will ever reach a certain target state.

What is missing from the reductionist view (at least, the simplistic one that I have given here) is the computational path. Even given the knowledge of the state and dynamical rules of the automaton, we cannot know the full computational path without simply simulating it ourselves. Therefore, the act of doing this computation is akin to a kind of knowledge discovery, or scientific investigation; albeit, one unlike inductive science, since we are attempting to resolve a totally deductive statement with totally deductive reasoning.

We can now make a statement about "emergent properties," in the spirit of Kolmogorov complexity.

If a property of a system is 'computationally irreducible' in an appropriate sense, then it is emergent.

By "irreducible", what I mean is that there is no algorithmic shortcut. For the kinds of problems we are considering here, computational complexity theory suggests that the best possible

[7] In particular, this problem is not necessarily in NP, but it should be in PSPACE. Knowing the precise complexity class is not important for the discussion here.

algorithm in general is brute-force simulation. This kind of computation is not “reducible” to any of the other pieces of knowledge that one might have about the system (in particular, its initial state or dynamical rules); it is a genuinely different kind of thing.

Of course, this is not a full definition of emergence. For example, we might also want to postulate that emergent properties are sufficiently global in nature. Nevertheless, in the case of cellular automata it seems that the global structure that might be described by a terminal state V is characterized not merely by the system rules and initial state, but by the full computational path needed to arrive at the state. Therefore, in this example, we can claim that arriving at state V is an emergent property.

Revisiting the definition of “emergence.”

The SEP defines “weak emergence” as follows.

> *Weak emergence affirms the reality of entities and features posited in the special sciences, while also affirming physicalism, the thesis that all natural phenomena are wholly constituted and completely metaphysically determined by fundamental physical phenomena, entailing that any fundamental-level physical effect has a purely fundamental physical cause.*
>
> *Special sciences describe non-ubiquitous, structured phenomena (e.g., plate tectonics, molecular interactions, cellular repair, and organismic development) and successfully predict their behavior through higher-level laws. Weak emergentists take the existence of such stable and distinctive phenomena, amenable to high-level but not low-level explanation, as reason to accept the taxonomic categories of the special sciences into our ontology of the natural world, no less real than the categories of a final, completed physics. On this view, there are molecules, cells, organisms, and minded creatures, and they do not reduce to—are not identical to—complex combinations of basic physical entities or features.*

This definition is suited to physical objects rather than abstract computational entities; adapting it to our discussion, I would define weak emergence as a commitment to the following:

a) There are phenomena “amenable to high-level but not low-level” explanations.
b) These phenomena have the same ontological status as their constituent parts.

Position (b) seems to not be at issue here, since we are discussing abstract mathematical/computational objects rather than physical ones.

As for (a), what is at issue is what exactly “high-level” versus “low-level” analysis means. As mentioned, the questions we are interested in can apply to any spatial scale. Unlike in physics, we are not distinguishing between big and small stuff. For example, in the state transition problem, formulating it as a purely local question does not reduce its computational difficulty.

(Localized) State Transition Problem: Given a cellular automaton with initial state U, where the cell at position (0, 0) is "off," will it ever reach a state where the cell at position (0, 0) is "on"?

Since large spatial scale is not the level at which things "emerge," what is? As I have argued, the relevant scale is that of computational difficulty. If a property of a system is maximally computationally difficult to deduce, then it is the most "emergent" in this sense. This kind of emergence might be called "scale-free emergence," since it does not depend on a specific spatial scale to emerge.

Note that this idea of emergence is not how computer scientists typically think of things! In the theory of algorithms, the existence of a global "structure" among a family of problem instances typically means that those problem instances can be solved with a fast algorithm. For example, it is known that constraint satisfaction problems which exhibit a global property known as *expansion* can be (approximately) solved in much faster than brute-force time. By contrast, constraint satisfaction problems with a lack of "structure" are believed to admit no such efficient algorithm.

However, while structure is sometimes the scaffolding upon which a fast algorithm relies, it can also be a barrier to *all* fast algorithms. In particular, the theory of pseudorandomness and the "structured vs pseudo-random" paradigm[8] demonstrate that hard instances of computational problems have their own kind of structure as well[9]. Moreover, unlike expansion or other algorithm-friendly structures, these structures are more "informationally dense" or "incompressible" by virtue of the fact that they *cannot* be solved by efficient algorithms.

Therefore, we can say that "structures" exist on every "scale" for abstract computational problems. In particular, computationally irreducible properties have their own kind of structure, and those can be said to be emergent. We can arrive at a tentative definition of weak computational emergence.

Weak computational emergence: Computationally irreducible problems have structures which are:

[8] Structure versus psuedorandomness is a very long and wonderful story. While Tao's paper is about combinatorics (and arithmetic combinatorics in particular), there is a long and still ongoing story about this interplay within computational complexity theory as well. One gem of this theory is the recent resolution of the 2-to-2 games conjecture, which gets us "halfway" to the Unique Games Conjecture. In that proof, a key step involves showing that pseudorandom subsets of the Grassman graph have near-perfect expansion. See this survey by Khot.

[9] This is a very debatable statement, of course. We are not in a position to make sweeping claims about the kinds of "structures" exhibited by hard computational problems, since the theory of computational complexity is still so young and so stuck on relatively simple questions such as P vs NP. Ultimately the word "structure" is ill-defined, and often its meaning changes as a mathematical field matures and introduces new concepts. If the motto of pseudorandomness is "randomness is in the eye of the beholder," then perhaps the motto of all mathematics is "structure is in the eye of the beholder."

a) *Amenable to "high-level" (computationally expensive) but not "low-level" (computationally inexpensive) explanations*
b) *These structures have the same ontological status as their constituent parts.*

Conclusion.

In this note, I have argued that the limits of computation pose problems for reductionism and certain theories of knowledge. My two main arguments are:

1) We should distinguish between deductive claims that are knowable with limited computational resources ("Type A"), and those that are so computationally expensive that our current physical knowledge rules out their resolution ("Type B").
2) Type B claims about cellular automata ask about properties that cannot be reduced to knowledge of individual system components - namely, the initial state of every cell and the rules for how cell states evolve.

I have also offered a tentative connection between computational irreducibility and "emergence," as well as a tentative definiton of "weak computational emergence." In the future I hope to build on this connection further, and in particular extend some ideas of Gregory Chaitin on algorithmic information theory and the philosophy of science.

Finally, some points that I have neglected in this note are:

(i) Are our brains just Turing machines (computers)? This is a complex issue that seems difficult to resolve absent a better understanding of the brain. With that said, I would be very surprised to find evidence that humans consistently outperform computers at the kinds of problems I'm discussing here, which are by design "purely computational," or "combinatorial." The entire point of NP problems is that they lack the kind of structure and patterns that allow our minds to make sense of complex data; indeed, the only thing to do is brute-force computation.

(ii) The global properties of NP problems besides cellular automata. Properly speaking, any so-called "complex system" (the nomenclature is unfortunate; these refer to systems arising in the study of 'complex systems theory' or 'complex adaptive systems', e.g. dynamic network models) has the properties of locality and global structure. Cellular automata are simply well-known and easy to describe.