A counting argument against triviality

Whit Schonbein
Department of Computer Science
University of New Mexico
email: schonbein@cs.unm.edu

March 10, 2017

Abstract

Computational triviality arguments attempt to show every physical system of sufficient complexity realizes every computational system. A standard response to these arguments is to note computational automata require counterfactual relationships between states, and this requirement imposes conditions on realization that do not permit full-blooded triviality; however, it is also commonplace to concede this response is insufficient. In this article I quantify the impact of the counterfactual response, showing it reduces the set of possibly realized computational systems to a vanishingly small fraction of those permitted without the restriction. I conclude by discussing the consequences of this result for metaphysical and epistemic triviality.

1 Introduction

Computational triviality arguments attempt to show every physical system of sufficient complexity realizes every computational system (Putnam, 1988; Searle, 1992; Godfrey-Smith, 2009). If sound, these arguments undermine the explanatory power of computational approaches to cognition (because there is nothing special about the type of computation performed by cognitive devices such as brains), or imply panpsychism (because many physical systems realize the type of computation performed by these devices).

A well-known response is to note the transition functions of computational automata specify *counterfactual* relations between computational states, and this imposes conditions on realization that curtail full-blooded triviality (Chalmers, 1996; Copeland, 1996). Unfortunately, most researchers view the counterfactual response as inadequate: while it may temper triviality, it still allows every physical system to realize a sufficiently large number of computational systems to cause metaphysical or epistemic difficulties for computational approaches to cognition (e.g., Piccinini, 2015, pp. 22-23).

To my knowledge, however, no attempt has been made to quantify the implications of the counterfactual response, and without such an analysis we have no clear grasp of its impact. In this article I show that requiring physical systems respect counterfactual relationships between computational states reduces the number of possibly realized computational systems to a minuscule fraction of those permitted without the restriction. I conclude by suggesting even if this result does not fully address concerns regarding metaphysical triviality, it helps explain why practices common to contemporary cognitive science are epistemically substantive.

2 Counting automata

The counterfactual response to triviality arguments asserts physical states cannot be 'repurposed' to play different causal roles in order to realize different automata. For example, if one deterministic finite automaton has the transition $\delta(q_1, a) = q_2$ (i.e., 'when in state q_1 and the input is a, transition to state q_2') while another has the transition $\delta(q_1, a) = q_3$, a physical system cannot realize both by transitioning from physical state r_1 to r_2 given input a on one occasion and from r_1 to r_3 given the same input on another (assuming the obvious mapping between physical and computational states). Given this restriction, any simultaneous realization of computational automata will be determined by whatever computational interpretations are consistent with causal structure being held constant. The question, then, is how many of these interpretations remain, relative to the original space of possibilities?

A reasonable strategy for approaching this question is to follow precedent by starting simple, e.g., with deterministic finite automata. A DFA comprises a finite set of states Q, a start state $q_0 \in Q$, a finite set of accept states $F \subseteq Q$, an input alphabet Σ, and a transition function δ from pairs of states and input symbols to states. The system processes an input $w \in \Sigma^*$ from left to right, beginning at q_0 and transitioning from state to state according to δ.[1]

A concrete example is given in figure 1, which depicts the minimal DFA for determining whether an input string of 0 or more instances of a has a length evenly divisible by three (i.e., if k is the length of the input, the system computes $k \equiv 0 \pmod 3$). Suppose physical system p implements this DFA: p has three states, $\{r_0, r_1, r_2\}$, begins in state r_0, and when perturbed by an external signal (which we will also designate 'a'), transitions according to the following transition rule: $\delta(r_0, a) = r_1, \delta(r_1, a) = r_2, \delta(r_2, a) = r_0$ (cf. Chalmers (1994)). Then, there appear to be four ways to 'reinterpret' p so that it realizes other DFA.

First, modify the symbols appearing in Q or Σ (and hence in δ). For instance, rather than label the physical states $\{r_0, r_1, r_2\}$, make them $\{A, B, C\}$; or, change the name given to the input signal from 'a' to '1'. However, since these modifications generate string-isomorphic variations on the original DFA,

[1]The Kleene star operator generates the set of all finite strings constructed from the members of a set; e.g., if $\Sigma = \{a, b\}$, then $\Sigma^* = \{\epsilon, a, b, aa, bb, ab, ba, aaa, bbb, \ldots\}$.

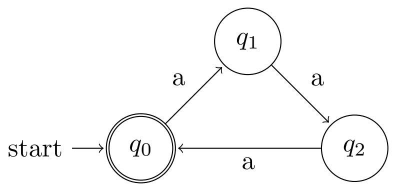

Figure 1: DFA d_1 for computing $k \equiv 0 \pmod 3$, where k is the length of the input string.

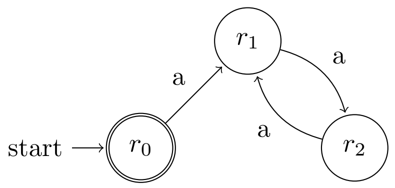

Figure 2: The structure of physical system p when reinterpreted as differing by one transition from d_1.

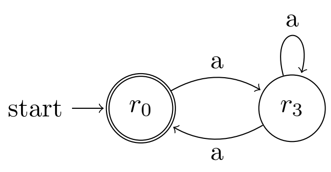

Figure 3: The structure of p when reinterpreted as having states r_1 and r_2 combined into a single state, r_3.

they do not qualify as distinct computational systems under normal accounts of computation (Hopcroft and Ullman, 1979).

Second, modify transitions between states. For example, change $\delta(r_2, a) = r_0$ to $\delta(r_2, a) = r_1$ (figure 2): now p no longer implements a DFA computing $k \equiv 0 \pmod 3$ but rather accepts just a single input, the empty string. Of course, this modification is inconsistent with the assumption the causal structure of p is fixed, i.e., that the transitions specified in d_1 are counterfactuals. Consequently, we cannot generate new interpretations of what p implements by modifying the organization described in δ.[2]

Third, add or subtract states. For example, figure 3 depicts a reinterpretation of p with states r_1 and r_2 combined into one 'super state'. The result is p now realizes a nondeterministic automaton accepting inputs of length $k = 0$ or $k \geq 2$. However, we've also disrupted the counterfactual structure of the original DFA: In d_1, an input of length three had no choice but to result in the system being in state r_0, but in the re-grouped version, it could be in either r_0 or r_3. So, once again, reinterpretation conflicts with the assumption that p has a causal structure mirroring the counterfactual structure of a computational system it realizes. (*Mutatis mutandis* for adding states.)

Fourth, change the accept states. Suppose we reinterpret p so that rather than having r_0 signify acceptance of an input signal, we make r_1 the accept state; now p defines the set of formulas whose length is $k \equiv 1 \pmod 3$. If we change it to r_2, we get the set of formulas whose length is $k \equiv 2 \pmod 3$. In other words, varying the accept state gives us three different DFA, each of which computes a distinct function. Furthermore, since more than one state can be an accept state, we must include all possible unions of these sets, adding another four DFA. Finally, if we omit an accept state altogether, we have a DFA computing the empty language. So, if we are allowed to vary the accept state, we get eight possible interpretations of the computational system realized by p.

At this point, I believe I've considered the obvious ways to reinterpret p given the assumption p implements DFA d_1. As this discussion shows, we cannot reinterpret what p computes by varying the number of states or the causal transitions between them, since in each case we violate the assumption these relations obey the relevant counterfactuals (as expressed in δ for d_1). However, nothing prohibits us from varying our interpretation of which states are accept states. Indeed, such variation appears consistent with how we might exploit the pre-existing causal structure of a physical system for computational purposes. For example, suppose we have a small furry mammal that reliably cycles through three observable states in response to gentle squeezes. Then, there is nothing in principle keeping us from using the causal structure of the animal to compute any of the eight functions identified above.

Generalizing, if the causal structure of a physical system with n states is held fixed, and the system realizes a DFA, then it computes 2^n possible functions, depending on how accept states are assigned. To assess whether 2^n qualifies as

[2]This is also why changing the start state is not an option; the entry to the computation is as much governed by counterfactual relations as the transitions between states within the system.

n	fixed transitions	variable transitions	ratio
1	2	2	100.00%
2	4	16	25.00%
3	8	216	3.70%
4	16	4,096	0.39%
5	32	100,000	0.032%
6	64	2,985,984	0.0021%
7	128	105,413,504	0.00012%
8	256	4,294,967,296	0.0000060%
9	512	198,359,290,368	0.00000026%

Table 1: Comparison of number of possible automata given n nodes and (i) fixed transitions (column 2) or (ii) variable transitions (column 3). Column 4 is the proportion of fixed to variable.

'many', we consider the ratio of automata realized by varying accept states to those made available when transitions can also be modified. A rough estimate of the latter is $(|\Sigma| - 1)n^n 2^n$, where $|\Sigma| \geq 2$ and $n \geq 1$.[3] The proportion is therefore:

$$\frac{2^n}{(|\Sigma| - 1)n^n 2^n} = \frac{1}{(|\Sigma| - 1)n^n} \leq \frac{1}{n^n}. \tag{1}$$

Equation 1 shows the number of automata realized by varying accept states is minuscule in comparison to the number made available by varying transitions: as n increases, the ratio rapidly approaches zero. Table 1 illustrates this fact by listing percentages for some values of n (using the more generous simplified equation $1/n^n$). When $n = 1$, the DFA realized by varying accepting states accounts for all of the DFA possible if transitions could also be varied. When $n = 2$, varying accepting states accounts for 25%, and when $n = 4$, varying accepting states accounts for less than 4%. By the time $n = 9$, the DFA realized by varying accepting states accounts for less than 0.00000030% of the possible DFA realizable if transitions could also be varied: if the number of 9-state DFA is a two-kilometer segment of freeway, then the subset realized by varying only accept states corresponds to the width of a strand of spider silk across the road.

This shows for even simple computational systems, the counterfactual response restricts the number of automata possibly realized by a physical system

[3]This estimate is calculated as follows. In order to guarantee every state is connected, there must be at least one path visiting each state, giving $(n-1)!$ possibilities (for sufficiently large Σ). The final transition in this path can go to any of the n nodes, adding another n possibilities. For sake of simplicity, assume this path uses the same input symbol for each transition, so this reduces to just a single possibility, which we drop from the calculation; this adjusts the estimate in favor of more automata being realized. The remaining transitions (for the remaining symbols) need not guarantee connectivity, so they can go to any of the n states, introducing approximately $(|\Sigma| - 1)n^n$ additional possibilities. For each of these, there are 2^n assignments of accept states, for a total of $(|\Sigma| - 1)n^n 2^n$.

to a vanishingly small subset of those realizable when its structure can be reinterpreted at will. Since the proportion of realized systems is far from 'large', it appears the counterfactual response is a promising reply to triviality arguments.

The obvious response to this analysis is to attempt to grow the ratio by increasing the numerator or decreasing the denominator. For example, someone might appeal to 'levels of analysis' to do the former: a single physical object can be simultaneously described at, e.g, the quantum level, the cellular level, the systems level, and at the level of organismic behavior. Each of these descriptions adds additional DFA realized by the physical system; with enough descriptions, the object realizes a sufficient number of DFA to trigger charges of triviality.

However, even in the best case scenario, it is unlikely this strategy will succeed. Suppose each level yields a description with the same number (n) of states, differing only in their causal structure. Then, for an n as small as 9, there would have to be more than one hundred and fifty million levels of analysis to account for just half of the DFA realizable when causal organization can be varied; the situation only worsens as n varies across levels. Therefore, appealing to descriptions at different levels is unlikely to make up the difference between 'minuscule' and 'many'.

A second objection is the argument significantly overestimates the number of automata possible when causal structure can be varied. Specifically, while some isomorphic DFA are accounted for in the estimate given in equation 1, not all are, so the estimate counts some automata more than once, making the proportion of simultaneously realized DFA seem much smaller than it actually is. In response, given the size of the gap between 2^n and n^n, it is unlikely the remaining isomorphic DFA are sufficient to make up the difference. For instance, assuming 50% qualifies as 'many', the estimate would have to be over-counting by $\frac{1}{2}(n^n - 2^{n+1})$, a value that rapidly goes to infinity as n increases.

A third objection is the estimates are based on considering DFA rather than more powerful automata; when other computational systems are taken into account, the ratio of systems realized when structure is fixed to those resulting from modifying that structure increases to the point where it is warranted to claim many of the possible automata are realized. This appears to not be the case. For example, under typical definitions of stack machines (Hopcroft and Ullman, 1979), the number of possible stack machines provided by varying accept states remains 2^n, even as the space of possible configurations grows; the same holds for standard definitions of Turing machines. Consequently, moving to more interesting automata only exacerbates the problem.

Finally, someone might object triviality does not arise from the number of simultaneously realized automata of a given type, but rather the number of realized automata of *different* types. For example, as Piccinini (2015, pp. 22-23) points out, the same causal structure can be described using many different computational formalisms, e.g., the Game of Life, C++, etc. However, even if this is the case, the preceding observations apply to each of these formalisms. Suppose p is described by token descriptions of computational types $C_1, C_2, \ldots, C_n$. Then, if the causal structure of p is held fixed, the possible instances of each type realized by p is restricted to a minuscule subset of the set of instances possible when causal structure can be varied. Summing over these subsets, we still get a vanishingly small number in comparison to the total space of possibilities.

3 Discussion

The counterfactual response is typically viewed as inadequate in the face of triviality arguments. However, when we count the number of automata permitted under that response, we find it succeeds in limiting token physical systems to realizing at most a vanishingly small fraction of the computational systems they could realize if their causal structure could be 'repurposed' as needed. Therefore, the counterfactual response is a *prima facie* promising reply to triviality arguments.

Someone might object this result nonetheless does not effectively handle the metaphysical issues raised by those arguments. Specifically, an 'absolutist' regarding the goals of an account of computational realization might hold that any satisfactory response to triviality arguments must reduce the number of possibly-realized computational systems to one, or to some number close to one. While the counterfactual response may eliminate the vast majority of computational systems from consideration, in comparison to any small constant, the number of remaining possibly-realized computational systems is still too high (2^n).

In response, one might question whether such demands on a satisfactory account of computational realization are reasonable. Regardless, even if they are, the failure to reduce the number of possibilities to a small constant in one fell swoop is not a reason to reject the counterfactual response; instead, we expect the appeal to counterfactuals – as well as other results concerning the relation between computational and physical descriptions – to work in tandem with additional considerations (e.g., teleological or mechanistic) to yield an account of computational realization for which pancomputationalism is not a problem. As a response to triviality claims, the import of the counterfactual response is that it does significant work towards addressing the problem, perhaps more than is typically granted in the literature.

The impact of the counterfactual response on epistemic triviality – i.e., the claim that attributing computational properties to a system is epistemically vacuous – is more immediate. On the one hand, the counting argument shows that identifying the causal structure of a physical system limits the system to possibly realizing a small fraction of computational systems when compared to the number available before that structure was known. Consequently, from an information-theoretic perspective, discovering causal structure is extremely informative. This helps explain why understanding the *actual* causal structure of physical cognitive systems is an important component to any computational account of cognition: by accurately characterizing that structure, we learn a lot about the sorts of computation cognitive systems may realize.

On the other hand, the argument suggests at least some computational hypotheses regarding cognition are empirically substantive: by identifying types of computation characteristic of cognition (e.g., systematicity, perhaps), we limit potential cognitive devices to those whose causal structure includes these types of computation in the sets of possibilities they support. As a simple example, if we vary the accept states of the DFA realized by the causal structures depicted in figures 1 and 2, there are only two languages computed by both: the empty language and the language containing all possible inputs. Consequently, if we assert (e.g., on the basis of behavioral evidence) a physical system comprising states and input-sensitive transitions computes $k \equiv 0 \pmod 3$, we can infer the former is a possible causal organization of that system while the latter is not. In general, identifying certain computations as central to cognitive processes can exclude some physical devices from the set of possible physical realizers of those computations, because they have the wrong causal structure. This helps explain why computational hypotheses concerning the status of cognition can be epistemically non-trivial.[4]

4 Addendum

An anonymous referee suggests the preceding argument makes two problematic assumptions that undermine the relevance of the results reported above:

1. The computational system being implemented is an outputless finite state automaton; and
2. The physical system has only finitely many states.

Rather than modify the main text, I address these concerns in this addendum.

Regarding (1), it is not clear to me from the referee comments exactly what the problem is supposed to be, but just as my argument extends to systems positing stacks, registers, etc., it easily covers outputs: let Δ be the output alphabet, and assume it has at least one member, ϵ, indicating 'no output'. If the system is a Mealy machine, outputs are associated with transitions between states, and there are $|\Delta|$ possible outputs. Therefore, assuming modifying outputs is consistent with holding counterfactual structure fixed, equation 1 becomes:

$$\frac{2^n(|\Delta|)}{(|\Sigma|-1)n^n 2^n(|\Delta|)} = \frac{1}{(|\Sigma|-1)n^n} \leq \frac{1}{n^n}. \tag{2}$$

In other words, when we factor outputs into the equation, they play a role in both counterfactual-preserving and non-counterfactual-preserving cases, so they cancel out. Similar outcomes hold for Moore machines, and for other automata involving outputs.

Regarding (2), in section 2 I discussed how the argument can be extended to Turing machines. A labeled digraph representation of any Turing machine computing a non-regular language has an infinite number of instantaneous states, i.e., to capture all of the possible configurations of such a computational system,

[4]I am grateful to Evan Dye for feedback on my calculations.

a physical system must provide an infinite number of physical states. Consequently, the claim my argument assumes the physical system has only finitely many states is incorrect.

I concede my argument assumes what a physical system can compute is determined by more than merely the number of states ascribed to it by a physical description; without this assumption, the counterfactual response is no response at all. This assumption may be controversial, but I believe it is well-supported by standard computational theory and practice. For example, if a physical description cannot branch between states it cannot compute the finite regular languages, if it cannot cycle it cannot compute the regular languages, and if it cannot count it cannot compute the context-free languages. This assumption is implicit in the example of realization given in section 2: to realize d_1, the physical system is described as having three states and the capacity to cycle between them (rather than merely requiring an infinite set of finite, non-branching, acyclic sequences of states, as in Putnam (1988)). Furthermore, it seems to me the assumption is present even in work on defining computation over $\mathbb{R}$, such as Blum, Shub, and Smale (1989), Moore (1998), and Tabor (2000). However, I agree this assumption may be too controversial to be made without explicit comment.

References

Blum, L., M. Shub, and S. Smale (1989). "On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions, and universal machines". In: *Bulletin of the American Mathematical Society* 21.1, pp. 1–46.

Chalmers, David J. (1994). "On implementing a computation". en. In: *Minds and Machines* 4.4, pp. 391–402.

— (1996). "Does a rock implement every finite-state automaton?" In: *Synthese* 108.3, pp. 309–333.

Copeland, B. Jack (1996). "What is computation?" In: *Synthese* 108, pp. 335–359.

Godfrey-Smith, Peter (2009). "Triviality arguments against functionalism". In: *Philosophical Studies* 145, pp. 273–295.

Hopcroft, John E. and Jeffrey D. Ullman (1979). *Introduction to automata theory, languages, and computation*. Reading, MA: Addison-Wesley.

Moore, Christopher (1998). "Dynamical recognizers: real-time language recognition by analog computers". In: *Theoretical Computer Science* 201, pp. 99–136.

Piccinini, Gualtiero (2015). *Physical Computation: A Mechanistic Account*. en. OUP Oxford. ISBN: 978-0-19-163342-3.

Putnam, Hilary (1988). *Representation and Reality*. Cambridge: MIT Press.

Searle, John (1992). *The Rediscovery of the Mind*. Cambridge, MA: MIT Press.

Tabor, Whitney (2000). "Fractal encoding of context-free grammars in connectionist networks". In: *Expert Systems* 17.1, pp. 41–56.