Chalmers on Implementation and Computational Sufficiency*

J. Brendan Ritchie

University of Maryland, College Park
britchie@umd.edu

Chalmers (2011) argues for the following two principles: computational sufficiency and computational explanation. In this commentary I present two criticisms of Chalmers' argument for the principle of computational sufficiency, which states that implementing the appropriate kind of computational structure suffices for possessing mentality. First, Chalmers only establishes that a system has its mental properties in virtue of the computations it performs in the trivial sense that any physical system can be *described* computationally to some arbitrary level of detail; further argumentation is required to show that the causal topology relevant to possessing a mind actually *implements* computations. Second, Chalmers' account rules out plausible cases of implementation due to its requirement of an isomorphism between the state-types of a computation and the physical system implementing the computation.

Key words: *Computation, implementation, mental properties, causal topology, explanation*

1. Introduction

Chalmers (2011) attempts to defend the following two principles:

> *Computational Sufficiency*: the appropriate kind of "computational structure" is sufficient for mentality.
> *Computational Explanation*: computation is a general framework for explaining cognition.

*I would like to thank Deepak Mirchandani, Gualtiero Piccinini, and an anonymous referee for helpful comments on an earlier draft of this paper.

Together these principles are intended to constitute the foundational role of computation in artificial intelligence and cognitive science. I am inclined to agree with both principles, as Chalmers *initially* states them.[1] However, in what follows I raise two criticisms of Chalmers' argument for computational sufficiency (CS).

First, Chalmers claims that mental properties are organizationally invariant with respect to the causal topology of a physical system, and that a causal topology can be "described," or "abstracted," as having the kind of grouping of physical states required by his account of implementation. However, Chalmers establishes the existence of such groupings only in the trivial sense that any physical system can be *described* computationally to some arbitrary level of detail; further argumentation is required to show that the causal topology relevant to possessing a mind actually implements computations. Second, Chalmers' account of implementation requires that the states of a physical system can be "grouped" so that they mirror the states of the computation(s) the system implements; that is, there is an isomorphism between formal and physical state-types. However, if the causal and the computational structural complexity of a physical system come apart in such a way that there is no isomorphism between formal and physical state-types, then perhaps only *some* implementations of the relevant computations suffice for mentality. Thus, further argumentation is required to show that: (i) the causal topology relevant to mentality in fact implements computations, and (ii) computations mirror those structural elements of causal topologies that are relevant to possessing a mind.

2. From Implementation to Computational Sufficiency

As Chalmers points out, the foundational role of computation in cognitive science and artificial intelligence requires clarifying the nature of computation.[2] While the mathematical theory of computation is well-characterized

[1] E.g. Chalmers (2011, p. 325) states computational sufficiency as follows: "the right kind of computational structure suffices for the possession of a mind". However, "computational structure" as Chalmers uses it here is ambiguous between different senses of "computation", as shall become clear below.

[2] I interpret Chalmers as primarily being concerned with digital (or rather,

(i.e. "abstract computation"), Chalmers claims that the interest in cognitive science and AI is with physical systems. What is required is an account of the bridge between abstract computation and certain physical systems, such that a physical system "realizes" an abstract computation, and an abstract computation "describes" the behavior of the physical computing system (i.e. the "concrete computations" of the system). Stated in this way, an account of implementation is of *both* these realizing and describing relations between abstract and concrete computation. With an account of implementation, Chalmers claims it can be shown that those properties of a physical system relevant to the system implementing some collection of computations are "precisely" the same properties in virtue of which the system has its mental properties (2011, p.327). And, if this connection can be shown, it would support CS.

Chalmers seeks to clarify the nature of computation in order to articulate and defend the foundational role of computation in cognitive science and AI. However, it is worth noting that there are two different notions of computation that are relevant here: abstract computation, as a mathematical formalism, and concrete computation, as a kind of physical causal process carried out by, for example, digital computers. One view is that cognitive science and AI are primarily interested in concrete computation. Evidence that Chalmers might be sympathetic to this view comes from two sources: first, his claim that these fields are interested in physical systems; and second, his claim that an account of implementation must be provided prior to justifying the foundational role of computation in cognitive science and AI. This seems to contrast with the influential view that it is primarily abstract computation that is central to cognitive science and AI, and how physical systems implement these computations is less central for explaining cognitive phenomena (e.g. Pylyshyn, 1984). However, Chalmers usage of "computation" seems to be restricted to abstract computation, though I will return to the topic of concrete computation in the concluding section. We now turn to Chalmers' account of implementation, and his argument for CS.

Informally, a physical system *P* implements a given computation *M* when

discrete) computation, as he later claims that such a framework can capture all important aspects of cognition (2011, p.351).

the causal structure of *P* “mirrors” the formal structure of *M*. In greater detail:

> *P* implements *M* when: (1) there exists a *grouping* of physical states of *P* into state-types; (2) a *one-to-one* mapping from formal states of *M* to the physical states types specified by (1); and (1) and (2) hold in such a way that: (3) the abstract state-transition relations between formal states of *M* are mapped *onto* physical state-types related by corresponding causal state-transition relations of *P*. (see Chalmers 2011, p. 328)

Call this Chalmers’ *informal* definition of implementation. Chalmers intends this definition to apply to a large class of categories, which he claims can be subsumed under the category of combinatorial-state automata (CSAs). CSAs are like finite-state automata, but with the important feature that the internal states of a CSA are constituted by vectors, where the individual elements are components of the overall state and the different possible values of the elements are possible sub-states of the overall state. *P* implements a CSA, *M*, when the following conditions are met:

> If there is a vectorization of internal states of *P* into components $[s^1, s^2, \ldots]$, and a mapping *f* from the substates s_j into corresponding substates S_j of *M*, along with similar vectorizations and mappings from inputs and outputs, such that for every state-transition rule $([I^1, \ldots, I^k],[S^1, S^2, \ldots]) \rightarrow ([S^1, S^2, \ldots], [O^1, \ldots, O^l])$ of *M*: if *P* is in internal state $[s^1, s^2, \ldots]$ and receiving input $[i^1, \ldots, i^n]$ which map to formal state and input $[S^1, S^2, \ldots]$ and $[I^1, \ldots, I^k]$ respectively, this reliability causes [*M*] to enter an internal state and produce an output that map to $[S^1, S^2, \ldots]$ and $[O^1, \ldots,$ Ol] respectively. (Chalmers 2011, p.331)

Call this the *CSA* definition. The CSA definition and the informal definition differ in their formulation in two ways that have been noted previously (see Scheutz, 2001): (i) only the informal definition makes explicit the requirement of a grouping of the physical states of a system into state-types, such that (ii) there is a bijection (one-to-one and onto) between the states of *P* and those of *M* (these definitions have the same mapping requirements if one assumes that the informal definition is also making reference to vectorized states).

Chalmers holds that the CSA definition can be utilized to account for other formalisms. For example "to develop an account of the implementation-conditions for a Turing machine, say, we need only *redescribe* the Turing machine as a CSA. Given this translation from the Turing machine formalism to the CSA formalism, *we can say* that a given Turing machine is implemented whenever the corresponding CSA is implemented" (2011, p.332, my italics). That is, we can describe a Turing Machine in such a way that its states can be appropriately grouped into internal states, with a vectorization of their elements and state-transitions such that the above conditions for implementation are satisfied.

Chalmers states that causal organization is the "nexus" between computation and the mind, for computation provides an "abstract specification of the causal organization of a system," and "if cognitive systems have their mental properties in virtue of their causal organization, and if that causal organization *can* be specified computationally, *then* the thesis of computational sufficiency is established" (2011, p.339, my italics). The truth of the antecedent of this claim should be straightforward for Chalmers, since in principle one "can" specify the causal organization of a system computationally (see below), and few would likely doubt that causal organization is essential for the instantiation of mental properties in a system. However, Chalmers spells out his reasoning here by introducing some further notions.

First, he introduces the idea of a "causal topology" of a physical system, which "represents the abstract causal organization of the system: that is, the pattern of interaction among parts of a system, abstracted away from the make-up of individual parts and from the way the causal connections are *implemented*" (2011, p.339, my italics). What the relevant notion of "implementation" is for a causal topology is unclear. If it is the same as that which applies for a CSA (which ultimately it would seem to be), then it is unclear how a "causal topology" is supposed to be different from an abstract computation, especially when Chalmers states that there are many "levels" of causal topologies in a system, but the one that is relevant to mentality will be at a fine enough grain to "determine the causation of behavior" (2011, p.339). If by "behavior" one means "output," then "causal topology" begins to sound like another name for abstract computation.[3]

Second, Chalmers defines a property *P* as being "organizationally invariant" with respect to a causal topology if for any causal topology preserving change to the system, *P* is also preserved (2011, p.339). I take this to be the idea that the property is a multiply realizable property of the causal topology. This interpretation accords with Chalmers' claim that mental properties are organizationally invariant. At some level of abstraction, this (functionalist) claim seems plausible, and I do not intend to dispute it here. When it comes to non-phenomenal, or "psychological" properties and states, this seems true under the claim that they can be defined by their causal-functional role in a cognitive system (for simplicity, I am setting aside Chalmers' arguments regarding phenomenal properties). Thus, following functionalism, Chalmers claims that systems with the same causal topology will share the same psychological properties; that is, as psychological properties are defined by their causal-functional roles within the structure of a system, and any two systems with identical causal topologies will have identical causal relations between states, it follows that any two systems with identical causal topologies will have identical psychological properties as well.

Given that mental properties are organizationally invariant, the final step for establishing CS simply depends on showing that organizationally invariant properties are "fixed by some computational structure"(2011, p.343). Now organizationally invariant properties depend for their instantiation on a pattern of causal interaction—a causal topology. And we can "straightforwardly abstract" a causal topology into a CSA description since "the parts of the system will correspond to elements of the CSA state-vector, and patterns of interaction will be expressed in the state-transition rules" (2011, p.343). Chalmers goes on to state that that the CSA formalism provides a "perfect" formalization of the idea of a causal topology. Support for this claim comes from the fact that "A CSA description specifies a division of a system into parts, a space of states for each part, and a pattern of interaction between these states. This is *precisely* what is constitutive of causal topology" (2011, p.344, my italics). However, this should be obvious given how

[3] For a related concern about how Chalmers characterizes the notion of causal topology, see Piccinini 2010, p.275, n.8.

"causal topology" was defined. As mentioned above, according to Chalmers, causal topology is an abstract "representation" and a computation is an abstract "specification" of the causal organization of a system.

According to Chalmers, this definition of implementation, when coupled with the notions of a causal topology and organizationally invariant properties, establishes CS, since "The fine-grained causal topology of a brain can be specified as a CSA. [And] Any implementation of that CSA will share that causal topology, and therefore will share organizationally invariant mental properties that arise from the brain" (2011, p.344). If one holds that CSAs provide "perfect" formalizations of causal topologies, then it would seem to follow that even the fine-grained structure of the brain can be specified using the CSA formalism. However, it strikes me as a bold empirical hypothesis that the causal structure of the brain can be specified as a CSA in any interesting way. This leads me to believe that something has gone amiss in Chalmers' argument.

3. Computational Sufficiency and Computational Description

I believe that Chalmers' argument for CS depends on some problematic inferences. However, before illustrating why I think this is so, I would like to point out that Chalmers seems to run together related, but distinct, claims all under the banner of "computational sufficiency". Chalmers seems to define CS in at least the following ways:

(CS_1) Having a particular computational *structure* suffices for possessing a mind.
(CS_2) *Implementing* a particular abstract computation suffices for possessing a mind.
(CS_3) Implementing a particular *CSA* suffices for possessing a mind.

CS_1, which is Chalmers' initial statement of CS, can be taken to refer to *concrete* computation, or is at least ambiguous about whether it refers to abstract or concrete "computational structure." I find CS_1 to be the most plausible of these principles, when interpreted in terms of concrete computation, and the most neutral (I return to this version of CS in the conclusion). However, I believe Chalmers *in fact* argues for CS_2, via CS_3. As I interpret it, Chalmers' argument for CS_3, and hence CS_2, has the following structure:

(1) Implementation is defined in terms of the informal definition and the CSA definition.[4]
(2) Other formalisms can be "redescribed" in terms of CSAs.
(3) Therefore, it follows that whenever, say, a Turing Machine is implemented, "we can say" that the corresponding CSA is implemented. (This establishes CSAs as the basis for a general account of implementation.)
(4) Mental states and properties are organizationally invariant with respect to the causal topology of a system (i.e. its abstract causal organization).
(5) We can "abstract" a system's causal topology into a CSA description. CSAs provide a "perfect" formalization of causal topologies.
(6) Therefore, any system that implements the relevant CSA will have the same causal topology.
(7) Therefore, implementing the relevant CSA will be sufficient for possessing a mind. (from (4) and (6))

I take (1)-(3) as being intended to establish that the CSA formalism can capture all categories of abstract computation. And if this is correct, then establishing (7) from (4)-(6) will not only establish CS_3 but also CS_2 as well.

I believe the above argument depends on making some problematic inferences, specifically from (2) to (3), and in defending (5). They are problematic, because if the first inference fails, then the CSA formalism cannot ground a general account of implementation, and if the second one fails, then (7) no longer follows. I take Chalmers' discussion of descriptions and abstraction to be equivalent to the claim that there are groupings that satisfy the conditions for implementation stated above, or rather, that we "can"

[4] These definitions have equivalent mapping conditions as mentioned earlier, if one assumes that the informal definition is, tacitly, conditions assuming vectorized states. Hence, I state both of them here (c.f. Scheutz, 2001).

provide such groupings. Here are what I take to be the problematic inferences:

> *(2) to (3)*: if we "can" describe a formal system, *F*, in terms of a CSA, *M*, it follows that whenever a physical system, *P*, implements *F*, it also implements *M*.
> *Defense of (5)*: Since we "can" abstract a CSA description, *M*, from a causal topology, *C*, such that *M* "specifies a division of a [*C*] into parts, a space of states for each part, and a pattern of interaction between these states," (2011, p.344) and this is "precisely" what a causal topology is, it follows that *M* is a "perfect" description of *C*.

The initial statements of implementation involved a mapping between an abstract computation and a grouping of a physical system's states and causal relations. As stated above, I take Chalmers talk of "description" and "abstraction" to refer to groupings of physical systems, which are specified independently of the mapping between a causal topology *C*, and a CSA, *M*. The problem as I see it is that Chalmers has failed to clearly distinguish between two kinds of groupings/descriptions, which we can call *modeling* descriptions and *explanatory* descriptions. My claim is that implementation depends on a mapping from abstract computations to explanatory descriptions/groupings; however, Chalmers establishes the existence of modeling descriptions only when making the above inferences. This distinction between kinds of descriptions can be elucidated with a related distinction made by Piccinini (2007a) between computational *models* and computational *explanations*.

One kind of computational description uses an abstract computation to provide a model of a physical system: the states of the physical system, *P*, are *represented* by the discrete states of an abstract computing system, *M* (e.g. a CSA or Turing Machine), and *P*'s causal state-transitions are represented by *M*'s abstract state-transitions, with *P*'s inputs, outputs, and internal states being similarly represented. As Piccinini (2007a) points out, it is not clear that all physical systems can be so represented, since many physical things fail to have discrete inputs, outputs, and internal states like digital computing systems. However, *M* might only *approximate*, rather than strictly map onto, the structure of *P*. For discrete computational systems, like a CSA, this requires *P* to be discretized in order for *M* to model *P*. Such a grouping, which allows for a computational system to model a physical system, is what I call a *modeling* description of a physical system: it is a grouping that approximates the actual structure of a physical system so that it can be represented by a computational model. It should be clear that the only way in which a modeling description would provide a grouping that meets the conditions for implementing a computation is a trivial one: anything that we can model by means of a computing system would therefore compute (i.e. implement some computation). Likewise, Piccinini (2007a) argues that if pancomputationalism amounts to the claim that any physical system can be modeled computationally to some arbitrary level of abstraction, it is likewise a trivial thesis. It would seem, however, that Chalmers disagrees, since he claims that pancomputationalism is not vacuous, and only would be so if one claimed that any system computes *every* computation. Instead, Chalmers claims that all physical systems merely implement a single-state FSA. However, we can again ask whether this amounts to the claim that all physical systems can be *described* as single-state FSAs.

The second kind of computation description, namely, computational explanation, is of a piece with how Chalmers understands computational explanation: the behavior of *P* is explained by a kind of process, namely, computation. One way of developing this is to say that a system computes only if it is a *computing mechanism*, i.e. a mechanism (composed of entities and activities with a particular organization, and producing some regular behavior; Machamer, Darden and Craver, 2000) that has the function of computing.[5] For example, different digital computing systems will have their own mechanistic explanations, but what they will have in common are certain functional characteristics: that their components or states constitute (strings of) digits, with other components that manipulate or store the strings, and that these operations are in accordance with rules that apply to all input strings, and only apply depending on the inputs to the system (c.f. Piccinini, 2007b). Many physical systems cannot be described in this more restrictive sense; they lack either a functional organization including com-

[5] For a defense and elaboration of this view see Piccinini, 2007b.

ponents that function as digits or states that function as inputs (e.g. a storm cloud). To a first approximation, an *explanatory* grouping is one that specifies the organization of a system in such a way that it explains the system's behavior (c.f. the notion of a mechanism *schema*; Machamer, Darden, and Craver, 2000). Not all ways of grouping the states and relations of a system are explanatory—some might even mischaracterize the causal structure of the system (c.f. Scheutz, 2001).

In making the above inferences, Chalmers at best argues for the existence of modeling descriptions. In the case of (2) to (3), there will be some way to vectorize a Turing Machine, but the fact that we can "describe" a Turing Machine, to some level of approximation, as a CSA, establishes (3) only in the trivial sense that if a physical system implements a Turing Machine (or rather, an approximation of a Turing Machine, with a finite tape), and some CSA models the Turing Machine, it therefore also models the physical system. Since part of Chalmers' motivation for adopting CSAs as his primary formalism was that it seems to provide a more accurate description of physical systems, it would seem that his argument fails by his own standards. After all, a CSA and a Turing Machine might compute the same function in extension, but not in intension; that is, they might have identical mappings from inputs to outputs, but differ in terms of the actual operations they specify. I take it that such a difference matters when we ask whether a physical system implements some computation, for if the operations are distinct then a physical system might only be able to implement one or the other set of operations because of its causal topology. If this is right, then arguing for CS_3 fails to suffice for establishing the more general CS_2, since we lack a formalism that is appropriate for defining implementation in general. Likewise, regarding the defense of (5), in so far as there is *some* level of approximation at which we can describe the causal topology of any physical system as consisting of, for example, discrete states and their state-transitions (from input and internal states to internal states and output states), it is true that a CSA provides a "perfect" description of the causal topology of a physical system *relative to* the approximate grouping. However, again, this will only be true in the trivial sense that a modeling description is being employed. If this is the sense of description that Chalmers is employing, then CS_2 is established only by being trivialized.

I do not think that Chalmers holds CS_2 to be a trivial principle. Again, given that he endorses the principle of computational explanation, it is fair to conclude that he thinks the mind has a computational structure in a non-trivial sense. However, his argument provides no reason to believe that there is an explanatory grouping of the causal topology with mental properties such that it implement CSAs. Perhaps the relevant causal topology of the brain is such that its fine-grained structure in fact consists of a number of discrete states, the internal structure of which can be explained as consisting of vectors with different discrete elements and values. But this is a strong empirical hypothesis, and given his resistance to burdening the computational foundations of cognitive science with such theses as symbolic computationalism (2011, p.353-354), I do not see why defending CS_2 should depend on such a hypothesis.

It seems that Chalmers argues for the existence of modeling descriptions in making the above inferences, but he assumes that he has established these as explanatory descriptions, which provide the groupings required for his claims about implementation, and hence CS_2. A diagnosis for this conflation is that Chalmers fails to clearly distinguish between functionalism, as the thesis that mental states are to be individuated functionally by their inputs, outputs, and relation to other mental states (i.e. their functional role), and computationalism (or “computational functionalism”), the view that the relevant functions and functional roles are computational. Piccinini (2004) has argued that this conflation is a result of endorsing two claims: (i) that all physical systems compute (i.e. pancomputationalism), and (ii) that the functional analysis of mental states is a kind of computational description.

I take Chalmers’ commitment to the organizational invariance of mental states and properties as implying a commitment to some version of functionalism (c.f. Chalmers’ [1996a, p.247] definition of functional organization). As noted earlier, his description of causal topology, which ostensibly also describes the functional organization of physical systems (in the case of systems for which a functional analysis is appropriate) is an abstract description of the causal organization of a physical system abstracting away from “implementation” details. So, Chalmers is clearly committed to pancomputationalism, and his description and discussion of causal topology suggests that he sees little difference between providing a functional analysis/description of causal topology, and providing a computational description.

However, I think we should reject both assumptions if we want the foundational role of computation in AI and cognitive science to be non-trivial. In fact, I worry that Chalmers also trivializes computational explanation. Chalmers claims that computation provides an "ideal language" for characterizing the causal organization (i.e. causal topology) of a system, and from this, he claims, computational explanation is supposed to follow (2011, p.345). However, all that he seems to have established is that computation provides a general framework for *modeling* behavior, unless one makes the two problematic assumptions discussed above.

4. Structural Complexity and Implementation

Recall that Chalmers ultimately wants to defend the claim that those properties of a physical system responsible for the system implementing some set of computations are exactly the same properties in virtue of which the system has mental properties (2011, p.327). So far I have been critical of whether Chalmers' argument establishes the claim that physical cognitive systems implement computations in the relevant sense, while remaining uncritical of his informal definition and CSA definition of implementation; that is, I have granted that if mental properties are organizationally invariant with respect to causal topologies, and these topologies indeed implement CSAs, then the above claim follows. In this last section, I want to raise a problem with these definitions, which calls into question the truth of the above claim.

Both informal and CSA definitions have in common the requirement (most explicit in the informal definition) that there be a bijection (one-to-one and onto) between the formal states of *M* and the physical states of *P*, such that for every state of *P* there is a corresponding state of *M*. In this respect, *M* and *P* have the same structural "complexity" regarding the number of states.[6] However, such a tight mapping between the causal and computa-

[6] In this section I use "complexity" to mean "structural complexity," or the number of state-types and state-transitions, for either the causal or computational

tional structural complexity of an abstract computation and the physical system that implements the computation rules out certain kinds of implementation. For example, this tight mapping rules out the possibility that one physical state implements two distinct computational states or that many physical states implement one computational state (Scheutz, 2001).

To account for such possible cases of implementation, Scheutz (2001) suggests that we reject the requirement of a bijection between states of *P* and *M* and define implementation in terms of *bisimilarity*, which requires that two systems have only the same sequences of state-transitions ("paths") but not the same number of states.[7] When defined using bisimilarity, there is a gap between the computational and causal structural complexity of a system: two systems might be equivalent in computational structure (they are bisimilar), but differ in their causal complexity (there is no bijection between their states). This, in turn, has consequences for CS_2, for it opens up the possibility that mental properties are organizationally invariant with respect to aspects of the causal topology other than its "paths."

For example, if mental properties are invariant with respect to the states of the causal topology, then a CSA will not reflect that part of the causal topology in virtue of which a system possesses mentality. However, it is possible that the causal paths are the only aspect of a causal topology for which mental properties are organizationally invariant. If so, Scheutz's revision of Chalmers' account is not problematic for CS_2. However, matters will again depend on the functional analysis of mental states and the precise nature of the causal topology of which they are organizationally invariant. Thus, Chalmers should either (1) argue that a single physical state cannot implement many computational states (a claim that I find to be implausible) or (2) explain why, if such an implementation is not possible, the paths of a causal topology are exhaustive of the causal structure relevant to possessing

structure of a physical system. This usage follows Scheutz (2001), and should be distinguished from the technical usage of "complexity" within computational complexity theory, where the term is used to denote the "difficulty" of different computational problems.

[7] Scheutz's (2001) suggested revision is to Chalmers' (1994, 1996b) definitions of implementation, which utilize FSAs as the main formalism. However, his suggestion is equally relevant here.

mentality (c.f. Scheutz, 2001).

5. Conclusion

Despite being critical of Chalmers' argument for CS_2, I think it can be mended. For Chalmers' case to succeed, he must provide a further argument to show that: (i) the causal topology relevant to mentality in fact implements computations (given his definitions) and (ii) computations mirror those structural elements of causal topologies that are relevant to possessing a mind. However, as stated earlier, I find CS_1 (when interpreted as a thesis about concrete computation) more plausible than CS_2 and CS_3 and more relevant to computation's foundational role in cognitive science and AI. Of course, a defense of CS_1 would require some account of concrete computation (i.e. computation by physical systems) and arguments for why it is the most important notion of computation for cognitive of science and AI. While there is no space here to defend the latter claim, I will end with a brief sketch of how one might defend CS_1.

One way to understand concrete *digital* computation is in line with Piccinini's (2007b) mechanistic account where concrete digital computation is constituted by a class of mechanisms (or "causal topologies") which meet certain structural and functional conditions. For example, central to Piccinini's mechanistic account is the function of manipulating strings of "digits," which is intended to be a concrete counterpart to the abstract notion of rules defined for operations over letters from a finite alphabet. While this account pertains solely to concrete digital computation, perhaps one could similarly try to provide a mechanistic account of concrete *neural* computations, whatever they may be. Relating this suggestion back to CS, one might then claim that what is sufficient for mentality is having a certain causal topology, or rather, a certain functional organization—an architecture that consists of a collection of mechanisms that have the function of computing. What kind of computations are performed by these mechanisms? I suspect information processing of a kind that qualifies as computation in at least a generic sense is part of the story (c.f. Piccinini and Scarantino, 2010). However, whatever the nature of the architecture it is likely unique to cognition and not easily assimilated into familiar formalisms, reflecting perhaps the *sui generis* nature of neural computation (c.f. Eliasmith, 2003; Piccinini and Bahar, forthcoming). As we do not as yet know what the relevant abstract computations are that the brain implements, it is hard to provide an account of implementation for these computations.

In conclusion, I do not think one needs to endorse the stronger claim that there is some abstract computation (or set of computations), the implementation of which is sufficient for mentality. We should not rule out the possibility that multiple systems might implement the same computation (equivalency of abstract computation) while possessing distinct causal topologies (distinctness of concrete computation), only some of which qualify as instantiating mental properties. Alternatively, we should not rule out the possibility that even concrete computational structure does not exhaustively describe the causal structure relevant to cognition. This is not to say that CS_2 is false; rather, I think that it puts a misplaced focus on abstract computation, when it could be claimed that it is concrete computation that is most relevant to justifying the foundational role of computation in cognitive science and AI.

References

Chalmers, D. 1994. On implementing a computation. *Minds and Machines* 4, 391-402.

Chalmers, D. 1996a. *The Conscious Mind*. New York: Oxford University Press.

Chalmers, D. 1996b. Does a rock implement every finite-state automaton? *Synthese* 108, 310-333.

Chalmers, D. 2011. A computational foundation for the study of cognition. *Journal of Cognitive Science* 12, 325-359.

Eliasmith, C. 2003. Moving beyond metaphors: Understanding the mind for what it is. *The Journal of Philosophy* 100, 493-520.

Machamer, P., Darden, L., & Craver, C. 2000. Thinking about mechanisms. *Philosophy of Science* 67, 1-25.

Piccinini, G. 2004. Functionalism, computationalism, and mental states. *Studies in History and Philosophy of Science* 35, 811-833.

Piccinini, G. 2007a. Computational modeling vs. computational explanation: is everything a Turing machine, and does it matter to the philosophy of mind? *Australasian Journal of Philosophy* 85, 93-115.

Piccinini, G. 2007b. Computing mechanisms. *Philosophy of Science* 74, 501-526.

Piccinini, G. 2008. Some neural networks compute, others don't. *Neural Networks* 21, 311-321.

Piccinini, G. 2010. The mind as neural software? Understanding functionalism, computationalism, and computational functionalism. *Philosophy and Phenomenological Research* 81, 269-311

Piccinini, G. & Bahar, S. (forthcoming). Neural computation and the computational theory of cognition.

Piccinini, G. and Scarantino, A. 2010. Computation vs. information processing: Why their difference matters to cognitive science. *Studies in History and Philosophy of Science* 41, 237-246.

Pylyshyn, Z. 1984. *Computation and Cognition*. Cambridge: MIT Press.

Scheutz, M. 2001. Computational versus causal complexity. *Minds and Machines* 11, 543-566.