Minds & Machines (2012) 22:137–148
DOI 10.1007/s11023-012-9280-4

# Computation, Implementation, Cognition

Oron Shagrir

Received: 31 January 2012 / Accepted: 18 May 2012 / Published online: 30 June 2012

**Abstract** Putnam (Representations and reality. MIT Press, Cambridge, 1988) and Searle (The rediscovery of the mind. MIT Press, Cambridge, 1992) famously argue that almost every physical system implements every finite computation. This universal implementation claim, if correct, puts at the risk of triviality certain functional and computational views of the mind. Several authors have offered theories of implementation that allegedly avoid the pitfalls of universal implementation. My aim in this paper is to suggest that these theories are still consistent with a weaker result, which is the nomological possibility of systems that simultaneously implement different complex automata. Elsewhere I (Shagrir in J Cogn Sci, 2012) argue that this simultaneous implementation result challenges a *computational sufficiency thesis* (articulated by Chalmers in J Cogn Sci, 2012). My focus here is on theories of implementation. After presenting the basic simultaneous implementation construction, I argue that these theories do not avoid the simultaneous implementation result. The conclusion is that the idea that the implementation of the right kind of automaton suffices for a possession of a mind is dubious.

**Keywords** Computational structure · Universal implementation · Simultaneous implementation · Mind · Grouping · Inputs and outputs

## Introduction

Hilary Putnam (1988) and John Searle (1992) contend that almost every physical system implements every finite computation.[1] This claim, if correct, puts at the risk

[1] An early version (from the 1970s) of this argument is attributed to Ian Hinckfuss (see in Cleland 2002). Hinckfuss points out that under a suitable categorization of states, a bucket of water sitting in the sun (The "Hinckfuss' pail") can be taken to implement the functional organization of a human agent.

O. Shagrir (✉)
Departments of Philosophy and Cognitive Science, The Hebrew University of Jerusalem,
91905 Jerusalem, Israel
e-mail: oron.shagrir@gmail.com

of triviality certain functional and computational views of the mind. One such view is the *computational sufficiency thesis* (CST), which is articulated by David Chalmers in “A Computational Foundation for the Study of Cognition” (Chalmers 2012). CST states that “the right kind of computational structure suffices for the possession of a mind, and for the possession of a wide variety of mental properties”. But if every physical system implements every computational structure (as Putnam and Searle claim), and if CST is true, then every physical system implements the computational structure that suffices for cognition (“a possession of a mind”). Hence, every physical system is a cognitive system. If CST is true, in other words, then rocks, chairs and planets have the kind of cognition that we do.

Chalmers (1996) and others have challenged the universal implementation claim; Chalmers offers, instead, a theory of implementation that allegedly avoids the pitfalls of universal implementation. My aim in this paper is to argue that the available theories of implementation are consistent with the nomological possibility of systems that simultaneously implement different complex automata; hence, if CST is true, each such system might simultaneously possess different minds. Elsewhere I argue that these cases undermine CST (Shagrir 2012). My focus here is on theories of implementation. After discussing Chalmers’s theory, I will examine whether other proposals can avoid this simultaneous implementation result.

## Putnam’s Argument

In the appendix to *Representation and Reality* (1988: 121–125) Putnam advances the claim that every physical system that satisfies minimal conditions implements every finite state automaton. He proves that every ordinary open system is a realization of every abstract finite automaton. Searle (1992) advances a somewhat similar claim, arguing that the wall behind him implements the Wordstar program. These claims clearly threaten CST. They imply that the human and the rock have the same functional organization. But if a functional organization of a certain complexity is sufficient for having a mind, as CST states, then the rock too should be deemed to have a mind. In fact, almost everything, given that it realizes this automaton, has a mind. Moreover, if Putnam’s theorem is true, then my brain simultaneously implements infinitely many different functional organizations, each constituting a different mind. It thus seems that I should simultaneously be endowed with all possible minds!

Putnam’s theorem applies to abstract finite state automata (FSA) without I/O (inputs/outputs). Take the FSA that runs through the state-sequence ABABABA in the time interval we want to simulate. Here A and B are the states of the FSA. Let us assume that a rock can realize this run in a 6-min interval, say from 12:00 to 12:06. Assume that the rock is in a maximal physical state $S_0$ at 12:00, $S_1$ at 12:01, and so forth (a maximal physical state being its total physical makeup specified in complete detail). Also assume that the states differ from each other (this is Putnam’s Principle of Noncyclical Behavior). Now let us define a physical state **a** as $S_0$ v $S_2$ v $S_4$ v $S_6$, and state **b** as $S_1$ v $S_3$ v $S_5$. The rock implements the FSA in the sense that the causal structure of the rock corresponds to the formal structure of the FSA. The physical state **a** corresponds to the logical state A, the physical state **b** corresponds to the logical B, and the causal transitions from **a** to **b** correspond to the computational transitions from A to B. A complete proof would require further elaboration, as well as a Principle of Physical Continuity. But it applies to any I/O-less FSA.

Putnam notes (1988: 124) that the proof cannot be immediately extended to FSA with I/O. If the I/O are functionally individuated, then they can be treated much like abstract internal states, and the extension is more natural. But if the I/O are specific kinds of physical organs, then the rock, which lacks motor or sensory organs of the required sort, cannot implement the automaton. The rock cannot implement a mind because it lacks the motor and sensory organs of thinking organisms. CST is in real trouble, however, if the difference between rocks and humans is exhausted by the I/O issue. After all, the whole point of CST is that the difference between thinking organisms and rocks is rooted in the complexity of functional organization. But if Putnam's argument is correct, the difference between humans and rocks is not rooted in the complexity of their respective internal computational arrangements (which turn out to be the same for humans and rocks), but in the kinds of I/O they can handle. This, it seems, is just a sort of behaviorism.

Indeed, Putnam points out (pp. 124–125) that reliance on physical I/O leads to behaviorism. Computationalism improves on behaviorism by taking into account not only I/O, but also the mediating algorithm, hence internal functional organization. Yet we accept behaviorism to be false because there are beings that have the same I/O dependencies as humans, but different mentalities (Block 1995). Computationalism holds that the reason they differ is that the "internal" relations among the logical states differ, that is, the implemented algorithm differs. But consider a physical entity with the right kind of I/O. What is true of this entity, on Putnam's theorem, is that it realizes all the possible algorithms that mediate the I/O. There can be no difference between the algorithms implemented by this entity and the algorithms that humans implement. In Putnam words: "In short, 'functionalism', if it were correct, would imply behaviorism! If it is true that to possess given mental states is simply to possess a certain 'functional organization', then it is also true that to possess given mental states is simply to possess certain behavior dispositions!" (1988: 124–125).

## Chalmers's Reply

Several people attempted to undermine Putnam's result, arguing that it takes more than Putnam allows to implement the functional organizations that are minds.[2] Chalmers (1996) provides a detailed counterargument along these lines. His contention is that there are constraints on the notion of implementation that are not taken into account by Putnam, and these constraints are not satisfied by rocks. One condition is that the state transitions of the implementing machine must be reliable

[2] See, for example, Block (1995), Chrisley (1994), Copeland (1996), and Melnyk (1996).

and counterfactual supporting. Another condition is that the causal structure of the physical object should mirror all the possible formal state transitions of the implemented FSA. In Putnam's proof, the rock implements only a *single run* (the transition from A to B and back), but not any other runs that might exist. If the FSA has other state transitions, e.g., C $\rightarrow$ D and D $\rightarrow$ C, these transitions should also be mirrored by the rock's dynamics.

It thus follows, according to Chalmers, that Putnam's proof applies to relatively simple kinds of automata, but not to the combinatorial state automata (CSA) that are more likely to be the minds that brains implement. A CSA is much like a FSA, except that it has a more complex, combinatorial internal structure. Each state is a combination of substates, and any state transition is sensitive to the combinatorial structure of the previous combined state: "CSAs are much less vulnerable to Putnam-style objections [than] FSAs. Unlike FSA implementations, CSA implementations are required to have complex internal structure and complex dependencies among their parts. For a complex CSA, the right sort of complex structure will be found in very few physical systems" (1996: 325). Chalmers concludes that brains, but not rocks, implement the complex CSA that is more likely to constitute a mind.

I do not contest the previous claims. Moreover, I agree with Chalmers about other claims he makes concerning implementation. One such claim is that every system implements an FSA: A rock implements a trivial one-state automaton. Another claim is that many physical systems typically implement more than one FSA. In particular, if a system implements a more complex FSA, it typically simultaneously implements simpler FSAs. A third claim is that only a few physical systems implement very complex CSAs. Particularly, very few physical systems implement a CSA that (*if* CST is true) suffices for cognition. I also agree with Chalmers on the point that the chance of an arbitrary physical system implementing such a CSA is very low.

However, I want to point out that one could construct systems that simultaneously implement different complex CSAs. Although I do not have a mathematical proof for this, I illustrate my claim with an example of simple physical systems that simultaneously implement different logical gates. The description I use is different from the one used by Chalmers. Chalmers describes automata in terms of "total states". I describe them in terms of gates (or "neural cells"), which are the basis of real digital computing. This description refers to parts of the machine and how they connect to one another. The two descriptions are known to be equivalent.[3]

## A Construction of Simultaneous Implementation[4]

Consider a physical system **P** that works as follows: It emits 5–10 volts if it receives voltages greater than 5 from each of the two input channels, and 0–5 volts

[3] Minsky (1967) demonstrates the equivalence between these descriptions (the proof is on pp. 55-58; the gate-description is presented in terms of McCulloch and Pitts "cells").

[4] A version of this example was first presented in an earlier paper (Shagrir 2001).

otherwise. Assigning ‘0’ to emission/reception of 0–5 volts and ‘1’ to emission/reception of 5–10 volts, the physical gate implements the logical *AND-gate*: ‘0’,‘0’ → ‘0’; ‘0’,‘1’ → ‘0’; ‘1’,‘0’ → ‘0’; ‘1’,‘1’ → ’1’.

Suppose it turns out that flip detectors of **P** are actually tri-stable. Imagine, for example, that **P** emits 5–10 volts if it receives voltages greater than 5 from each of the two input channels; 0–2.5 volts if it receives under 2.5 volts from each input channel; and 2.5–5 volts otherwise. Let us now assign the symbol ‘0’ to emission/reception of under 2.5 volts and ‘1’ to emission/reception of 2.5–10 volts. Under this assignment, **P** is now implementing the *OR-gate*: ‘0’,‘0’ → ‘0’; ‘0’,‘1’ → ‘1’; ‘1’,‘0’ → ‘1’; ‘1’,‘1’ → ‘1’.[5]

The result is that the very same physical system **P** simultaneously implements two distinct logical gates. The main difference from Putnam’s example is that this implementation is constructed through the very same physical properties of **P**, namely, its voltages. Another difference is that in the Putnam case there is an arbitrary mapping between different physical states and the same states of the automaton. In our case, we couple the same physical properties of **P** to the same inputs/outputs of the gate. The different implementations result from coupling different voltages (across implementations, not within implementations) to the same inputs/outputs. But within implementations, relative to the initial assignment, each logical gate reflects the causal structure of **P**. In this respect, we have a fairly standard implementation of logical gates, which satisfies the conditions set by Chalmers on implementation.

It is important to see that we could create other dual gates in a similar way. Consider a physical system **Q** that emits 5–10 volts if it receives over 5 volts from exactly one input channel and 0–5 volts otherwise. Under the assignment of ‘0’ to emission/reception of 0–5 volts and ‘1’ to emission/reception of 5–10 volts, **Q** implements an *XOR-gate*: ‘0’,‘0’ → ‘0’; ‘0’,’1’ → ‘1’; ‘1’,‘0’ → ‘1’; ‘1’, ‘1’ → ‘0’. Suppose, as before, that flip detectors of **Q** are tri-stable. Suppose that **Q** emits 5–10 volts if it receives voltages higher than 5 from exactly one input channel; 0–2.5 volts if it receives 5–10 volts from both input channels; and 2.5–5 volts otherwise. Assigning the symbol ‘0’ to emission/reception of under 2.5 volts and ‘1’ to emission/reception of 2.5–10 volts, **Q** is now implementing the *NAND-gate*: ‘0’,‘0’ → ‘1’; ‘0’,‘1’ → ‘1’; ‘1’,‘0’ → ‘1’; ‘1’,‘1’ → ’0’.

Another example is of a physical system **S**. This system emits 5–10 volts if it receives over 5 volts from each input channel and 0–5 volts otherwise. Under the assignment of ‘0’ to emission/reception of 0–5 volts and ‘1’ to emission/reception of 5–10 volts, **S** implements an *AND-gate*. As in the previous cases, it turns out that flip detectors of **S** are actually tri-stable. **S** emits 5–10 volts if it receives voltages higher than 5 from each input channel; 0–2.5 volts if it receives under 2.5 volts from at least one input channel; and 2.5–5 volts otherwise. Assigning the symbol ‘0’ to emission/reception of under 2.5 volts and ‘1’ to emission/reception of 2.5–10 volts, **S** is now implementing the *AND-gate* (again!). This **S** system is interesting in that the two implemented *AND-gates* might be in different “total states”, under the same physical conditions. If, for example, the inputs are 3 volts

[5] Sprevak (2010) presents this result more elegantly without invoking tri-stable flip-detectors.

from both input channels (and so the output is, say, 3 volts), then this very same *physical run* simultaneously implements the ‘0’,‘0’ → ’0’ mapping under the first implemented *AND-gate*, but the ‘1’,‘1’ → ‘1’ mapping under the second implemented *AND-gate*.

I do not claim that each simple physical system implements every logical gate. I also do not claim that they implement a complex CSA. The point I want to emphasize is that one can use these *physical* gates (i.e., **P**, **Q**, **S** and the like) as building blocks of physical systems that simultaneously implement very complex automata. To illustrate this point, let us consider a computation that has a series of four *AND-gates* at one point. One way to implement this computation is with a sequence of four physical systems, **P**, each of which implements the *AND-gate*. But **P** also implements, simultaneously, the *OR-gate*. Thus the sequence of the four physical, **P**, systems, simultaneously implements another computation, that of a series of *OR-gates*. One can point out that while I can switch my “interpretation” from *AND* to *OR*, I cannot take those very same **P** gates and treat them as alternating *AND-OR-AND-OR*. Switching the interpretation requires that I will be consistent with respect to the individuation of the voltages. Either I assign ‘1’ to receiving/emitting of more than 2.5 volts for all gates, or to receiving/emitting of more than 5 volts for all gates. I cannot decide to switch my interpretations across gates. This observation is quite correct. But it does not imply that I cannot simultaneously implement a computation that consists of a series of four *AND* operations, *and* a computation that consists of a series of alternating, *AND-OR-AND-OR*, operations. I could use a series of different *physical* systems: **S**–**P**–**S**–**P**. This series simultaneously implements both computations. When assigning ‘1’ to receiving/emitting of more than 5 volts (and ‘0’ otherwise) the system computes a series of four *AND* operations. When assigning ‘1’ to receiving/emitting of more than 2.5 volts (and ‘0’ otherwise) the system computes a series of *AND-OR-AND-OR* operations.

The more general point is that if you want to implement a zillion-gate automaton, you can use these and other gates to implement another automaton of “the same degree”, where “the same degree” means the same number of logical gates. Automata that include finite and infinite memory (as in Turing machines) are no obstacle to this result; we can individuate the 0 s and 1 s in the memory the same way we individuate the inputs and outputs. Thus “same degree” also means the “same amount of memory”. I do not claim that this ability is shared by every physical system. We can assume that physical systems often implement only one automaton of the same degree. It might also be the case that, technologically speaking, it will be ineffective and very costly to use these tri-stable gates. The point is rather philosophical: If implementing some CSA suffices for the possession of a mind, then I can construct a physical system that simultaneously implements this CSA *and* another CSA of the same degree. I can also construct two instances of the same CSA, each of which is in a different total state. These constructions are not just a logical or metaphysical possibility. They are nomological possibilities; in all likelihood, they are also technologically feasible.

## Implications to CST

If Putnam's theorem is valid, then any organism implements every finite automaton, and perhaps even many other kinds of automata. It therefore seems to undermine CST in two ways: (a) A rock implements the same functional organization I do, but the rock lacks mentality; (b) My brain implements many functional organizations, each of which normally constitutes an independent mind, even though I have but a single mind. Chalmers's theory of implementation removes the first worry, of a rock implementing the same functional organization as I do. But, based on the construction above, I now want to claim that the reply does not remove the second worry, e.g., of my brain simultaneously implementing many complex independent CSAs, each of which embodies an independent mind.

Let us assume (*by reductio*) that computational structure suffices for possessing a mind (at least in the sense that a mind supervenes on computational structure). Then two creatures that implement the same structure cannot differ in cognition, namely, one having a mind and the other not. Presumably, having a mind requires the implementation of a "very complex" CSA, perhaps a potentially infinite machine. This automaton might be equipped with more features: The ability to solve (or approximate a solution for) certain problems in polynomial-bounded time; some universality features; the ability to support compositionality and productivity; and so on. Let us call an automaton that underlies minds a MIND.

The next step in the argument is the claim that CST is consistent with the nomological possibility of certain scenarios. I have no proof for this consistency claim, partly because we do not know what it takes to be a MIND. But I think that the friends of CST should at least address these scenarios. One scenario is that of a physical system that simultaneously implements different MINDS, and, hence, possesses two different minds. We can assume that the two MINDS are of the same degree of complexity (if not, then we use the number of gates, memory, etc., required for the more complex automaton). I do not claim that any arbitrary system can do this. I also do not claim that any "complex" CSA suffices for a mind. My claim, rather, is that we can construct ("can" in the sense of nomological possibility) a physical system out of (physical) gates that are tri-stable, each of which implements two different automata that are of the same degree of MIND. Moreover, using these tri-stable gates, we can construct a physical system that implements a MIND and another CSA with the same complexity. The tri-stable physical gates we choose are just the ones with the Boolean functions needed to implement MIND. Indeed, it seems that it is nomologically possible to simultaneously implement a MIND and many different such CSAs through *n*-i stable gates.

The second scenario is that of a physical system that implements two instances of the same MIND. We can do this by planting tri-stable gates that implement the same Boolean function (like the system **S** described above). The physical system simultaneously implements two different total states of the same MIND. Assuming that phenomenal states nomologically (though *not* logically) supervene on computational structure (Chalmers 1995), this system simultaneously possesses the experience that the wall in front of it is wholly red and the experience that the wall in front of it is wholly blue. The claim is *not* that the same mind simultaneously possesses different, even contradictory, experiences. The claim is that the very same physical system simultaneously possesses two minds (perhaps two instances of the same mind!) that have different experiences. But one mind (or instance of the same mind) does not have to be aware of the experiences of the other.

To summarize, I have argued that if Chalmers's theory of implementation is correct, then it is nomologically possible to simultaneously implement more than one CSA (including itself) of the same degree. Thus if CST is true, there is a nomologically possible physical system that simultaneously implements a MIND and at least another CSA that might be a mind too.

But what follows from this? Chalmers addresses this possibility, saying that "a given physical hunk of matter can be associated with more than one mind" (1996: 332), yet he does not find it too troubling. He says that "there may even be instances in which a single system implements two independent automata, both of which fall into the class sufficient for the embodiment of a mind. A sufficiently powerful future computer running two complex AI programs simultaneously might do this, for example" (1996: 332). Elsewhere (Shagrir 2012) I argue that this result makes CST less plausible. It brings with it a host of epistemological problems (since the "other" minds are deluded and epiphenomenal); it threatens the idea that the mental-on-physical supervenience is a dependence relation, and it undermines the claim that CST provides conceptual foundations for the computational science of the mind. My aim here is to consider two attempts to avoid this multiple-minds result altogether by putting more constraints on the notion of implementation.

## Grouping

Matthias Scheutz (2001) advances a rigorous criticism of Chalmers's theory of implementation. His main line is that the more important constraint on implementation, other than being counterfactual supportive, is the grouping of physical states.[6] Scheutz argues that as long as no constraints are imposed on the groupings of physical properties that form the implementing states, Putnam's theorem might be rehabilitated: "If no restrictions are imposed on groupings of physical states, then simple, finite, deterministic physical systems…can possibly be seen to implement complex, infinite, and non-deterministic computations" (2001: 551). Indeed, Moore (1990) shows that even a universal Turing machine can be embedded in the motion of a single particle moving in space, bouncing between parabolic and linear mirrors like an ideal billiard ball. The infinite tape and table of instructions can be embedded in the binary development of the coordinates of the particle's position. It might even turn out that if the groupings are defined over the quantum makeup of a rock, the rock implements a complex CSA that is, arguably, constitutive of mind.

[6] See also Scheutz (1999); for a related discussion see Melnyk (1996) and Godfrey-Smith (2009). Godfrey-Smith mainly addresses triviality arguments that group different types of physical states into one "implementing" type via disjunction. He thus suggests that the "substate variables map to independent parts of the realizing system", and that "the parts' microstates grouped into coarse-grained categories be physically similar" (2009: 292). I concur with these modifications, but they do not undermine my construction, which does not proceed through disjunctive grouping.

Scheutz (2001, Sect. 4) offers a revised theory of implementation. I will not get into the full details of the theory here, but just mention the two main additional constraints it imposes. One is that the implementation is always relative to a fixed canonical physical theory (e.g., circuit theory), a theory on which the grouping into physical types is already given. Thus a rock might implement a complex CSA relative to the grouping of quantum mechanics, but it allegedly implements only very simple FSAs relative to the grouping of some macro-physical theory. The second constraint is related to the first. Let *S* be a physical system, and *GS* a grouping of its *physical* states into types (note that the grouping is made at the level of the *physical* theory). Let $M^{GS}$ be the characteristic automaton with respect to *GS*; this is the automaton that "*truly captures and reflects the entire causal structure of the physical system for the given set of physical states* [i.e. relative to the grouping GS] by retaining the mere structural properties of the physical system while abstracting of the particular physical qualities of the physical states" (Scheutz 2001: 551). Finally, we will say that *S* implements an automaton *M* (relative to *GS*) just in case M is at most as complex as $M^{GS}$; roughly, M is at most as complex as $M^{GS}$ if it is implemented by every physical system that implements $M^{GS}$. This condition excludes counterexamples to Chalmers's theory, in which the unwarranted groupings lead to wrong implementations.

The first condition is highly useful for the purposes of computer science, but might be of little help to functionalists and computationalists who hold to a version of CST. As Putnam repeatedly notes, the functionalist has to pick out one physical grouping and not another without appealing to any semantic or intentional traits. Given that functionalism is a reductive theory, it would be unfair to describe humans but not rocks in terms of the pertinent characteristic automaton only because humans are deemed to have minds, and rocks are not. To do so would be totally circular. The computer scientist is in a very different position. To fix the canonical grouping, the computer scientist can and does appeal to traits like goals, purposes, and desiderata such as easy-to-build and user-friendly interfaces. The job of the computer scientist is not, and has never been, akin to that of the functionalist, namely, to provide a reductive theory of content.

The second condition, about complexity, is indeed critical. But it does not seem to undermine my construction. I concede that every physical system described in my construction implements a deeper, "more complex", characteristic automaton that entails the simultaneous implementation of the "less complex" automata. Thus the system **P** in fact implements a tri-value logical gate that entails both the *AND-gate* and the *OR-gate*. The same goes for the implementation of MIND: there is a "more complex" characteristic automaton that entails the implementations of the "less complex" MIND. My point, rather, is that these constructions satisfy the complexity condition. Every system that implements the tri-value function (implemented by **P**) also implements the *AND-gate* and the *OR-gate*. The same goes for the more complex constructions. I can construct a physical system whose characteristic automaton entails the implementations of MIND (hence, a mind) and perhaps another—any other—automaton of the same degree.[7]

[7] One could stipulate that minds are to be associated with characteristic automata but this condition is obviously ad hoc. This will entail that one loses her mind if some of her neurons turns out to be tri-stable rather than bi-stable.

The upshot, then, is that Scheutz's revised notion of implementation is, as far as I can tell, consistent with my argument against CST. I should emphasize that I do not hold this result against Scheutz. I think that his revised notion is an important improvement on Chalmers's theory; what I think should be rejected, rather, is CST. Retaining CST, it seems, requires even further constraints on the notion of implementation.

## Inputs and Outputs

Another way to strengthen the notion of implementation is to put more restrictions on inputs and outputs (I/O). Chalmers (2012) himself says that "it is generally useful to put restrictions on the way that inputs and outputs to the system map onto inputs and outputs of the FSA". The question is how to specify I/O and where to locate them (i.e., proximal or distal). Presumably, an intentional specification is out of the question; it undermines the main goal of CST which is to account for minds in non-semantic and non-intentional terms.[8] Another option is to characterize the more distal I/O in functional (abstract) terms. Thus if (say) the output of 2.5–10 volts is plugged to a movement of the arm, and the output of 0–2.5 volts to no-movement, we can say that one automata is implemented and not another. My reply (to reiterate a point made in the "Putnam's Argument" section) is that this option helps to exclude the implementations in the case discussed above, but is of no help at all in the general case. For there is yet another system in which the output of 2.5–5 is plugged to (physical) low-movement, and the output of 0–2.5 is plugged to no-movement. In this case we can individuate movement (which is just '1') either to high movement or to low-movement-plus-high-movement. In short, there seems to be no principled difference between the syntactic typing of internal and external events. It might help in certain cases, but it does not greatly change the end result about simultaneous implementation. The construction is just extended to include the wider abstract inputs and outputs, which can be individuated in different ways.

A third option is a physical (including chemical and biological) specification of I/O. We can tie the implementation to a specific proximal I/O, say, the currents of 2.5–10 volts, or to something more distal, say, physical motor movement.[9] But this move has its problems too (for CST). One is that this proposal runs against the idea that "computations are specified syntactically" (Chalmers 2012); for they now depend on specific *physical* inputs and outputs. Chalmers's idea, more specifically, is that computational properties are fixed by the causal topology of the system, which means that while it is important to implement an abstract automaton in some physical properties, it is not important which physical properties implement the automaton. Moreover, this move to physical I/O has the risk of turning the computational views into forms of behaviorism, in that we shift the burden of individuative work onto I/O.[10] Another problem is that the proposal has undesired

[8] Thus Chalmers (2012) writes: "It will be noted that nothing in my account of computation and implementation invokes any semantic considerations, such as the representational content of internal states. This is precisely as it should be: computations are specified syntactically, not semantically".

[9] Godfrey-Smith (2009) distinguishes between a broad and a narrow construal of inputs and outputs.

consequences about the typing of computations and minds. Let us say that implementation is tied to the physics of the motor movements. The result is that a robot whose arms are made of metal can never be computationally (hence, mentally) equivalent to a human whose arms are made out of biological material. The reason being that if the physics of inputs and outputs are different then the computational (hence, mental) types are different too.[11]

Another option is advanced by Gualtiero Piccinini (2008). According to this view, we need to take into account the functional task in which computation is embedded, and this task may depend on the interaction between the computing mechanism and its context, e.g., on "which external events cause certain internal events" (2008: 220). Specifically, in order to specify the mechanism's I/O, we have to know how they interact with their environment. Piccinini adds that the environment does not have to be very wide. In artificial computing systems the relevant functional properties might be input (e.g., keyboard) and output (e.g., display) devices. In the cognitive case it might be sensory receptors and muscle fibers. This might be enough to determine whether and which computation is being performed.[12] This proposal has its merits, but something has to go. There are two ways to understand it. On one understanding, a physical system might simultaneously implement different complex automata. But whether or not an implemented automaton suffices for mind depends on its causal interaction with external events. This is a viable position, but it relinquishes CST. The implemented automaton alone does not suffice for mind; its causal interactions with external events are essential too.[13] On another understanding, a system does not simultaneously implement different complex automata. It implements only the automaton that causally interacts with external events. The external events, in other words, are part of the implemented automaton. In this case we are back to square one. If they are individuated functionally, we can extend the argument to the environment. We can have a construction in which a system simultaneously performs different functional tasks. If the external events are individuated physically, we tie computational implementation and individuation to specific physical properties.

There might be other viable options to constrain I/O. But the discussion indicates that finding the adequate constraints on I/O is no easy task.

## Summary

I have proposed a construction of physical systems that simultaneously implement a variety of abstract complex automata. Based on this construction I argued that if

[10] This was suggested by Putnam (see the discussion in the "Putnam's Argument" section). Godfrey-Smith (2009) advances an argument that aims to show that merely appealing to physical I/O does not block (other) triviality results.

[11] There might be other physical properties that the robot metal arm and the biological human arm share; see Block(1978) for general discussion.

[12] Something along these lines has been proposed by Scheutz (private communication).

[13] It should be noted, however, that Piccinini does not aim to support CST. His proposal targets claims about individuation.

CST is true, then there are nomologically possible physical systems that simultaneously possess different automata that might give rise to minds. Elsewhere I have argued that this result presents a serious challenge to CST (Shagrir 2012). Here I argue that my construction is consistent with several notions of implementation (which is not to deny that it might be inconsistent with other to-be-revised notions of implementation). My conclusion is that the idea that an implementation of the right kind of automaton suffices for a possession of a mind is dubious.

**Acknowledgments** I am grateful to Darren Abramson, Matthias Scheutz, Eli Dresner, Ilan Finkelstein, and three anonymous referees, for their comments, suggestions and corrections. The paper was presented in the *Philosophy and Theory of Artificial Intelligence (PT-AI 2011)* conference in Thessaloniki (October 2011). I am thankful to the participants for the lively and thorough discussion. This research was supported by The Israel Science Foundation, grant 1509/11.

## References

Block, N. (1978). Troubles with functionalism. In: W. Savage (ed.) *Perception and Cognition: Issues in the foundations of psychology*, Minnesota Studies in the Philosophy of Science (Vol. 9, pp. 261–325). Minneapolis: University of Minnesota Press.

Block, N. (1995). The mind as the software of the brain. In D. Osherson, L. Gleitman, S. Kosslyn, E. Smith, & S. Sternberg (Eds.), *An invitation to cognitive science, volume 3: Thinking* (2nd ed., pp. 377–425). Cambridge, MA: MIT Press.

Chalmers, J. D. (1995). Absent qualia, fading qualia, dancing qualia. In T. Metzinger (Ed.), *Conscious experience*. Paderborn: Ferdinand-Schoningh.

Chalmers, J. D. (1996). Does a rock implement every finite-state automaton? *Synthese, 108*, 309–333.

Chalmers, J. D. (2012). A computational foundation for the study of cognition. *The Journal of Cognitive Science*, forthcoming.

Chrisley, L. R. (1994). Why everything doesn't realize every computation. *Minds and Machines, 4*, 403–420.

Cleland, E. C. (2002). On effective procedures. *Minds and Machines, 12*, 159–179.

Copeland, B. J. (1996). What is computation? *Synthese, 108*, 335–359.

Godfrey-Smith, P. (2009). Triviality arguments against functionalism. *Philosophical Studies, 145*, 273–295.

Melnyk, A. (1996). Searle's abstract argument against strong AI. *Synthese, 108*, 391–419.

Minsky, M. (1967). *Computation: Finite and infinite machines*. Englewood Cliffs, NJ: Prentice-Hall.

Moore, C. (1990). Unpredictability and undecidability in dynamical systems. *Physical Review Letters, 64*, 2354–2357.

Piccinini, G. (2008). Computation without representation. *Philosophical Studies, 137*, 205–241.

Putnam, H. (1988). *Representations and reality*. Cambridge, MA: MIT Press.

Scheutz, M. (1999). When physical systems realize functions. *Minds and Machines, 9*, 161–196.

Scheutz, M. (2001). Causal vs. computational complexity? *Minds and Machines, 11*, 534–566.

Searle, J. (1992). *The rediscovery of the mind*. Cambridge, MA: MIT Press.

Shagrir, O. (2001). Content, computation and externalism. *Mind, 110*, 369–400.

Shagrir, O. (2012) Can a brain possess two minds? *Journal of Cognitive Science*, forthcoming.

Sprevak, M. (2010). Computation, individuation, and the received view on representation. *Studies in History and Philosophy of Science, 41*, 260–270.