Searle's Wall

James Blackmon

1.

John Searle has famously argued against computationalism, the view that mental states are computational states, charging that the view attributes minds to too many things. His Chinese Room argument is well known and has received much attention. But Searle has a more radical argument that can be used to this end, one I will call *Searle’s Wall*.[1] (Searle, 1992)

> On the standard textbook definition of computation, it is hard to see how to avoid the following results:
>
> 1. For any object there is some description of that object such that under that description the object is a digital computer.
>
> 2. For any program and for any sufficiently complex object, there is some description of the object under which it is implementing the program. Thus for example the wall behind my back is right now implementing the Wordstar program, because there is some pattern of molecule movements that is isomorphic with the formal structure of Wordstar. But if the wall is implementing Wordstar, then if it is a big enough wall it is implementing any program, including any program implemented in the brain. (pp. 208-9)

We can see how Searle's Wall, if successful, would be a problem for any view according to which the implementation of certain computations is sufficient for the realization or instantiation of mental properties. The alleged result is be among the most literal and drastic forms of panpsychism: Walls are thinking thoughts just as we are. In fact, everything thinks so long as it meets a basic complexity requirement, and this requirement can be met simply by being big enough.[2] And of course the problem extends beyond computationalism, threatening to undermine the entire cognitivist program as well as the widely held notion that program implementation is a non-trivial physical phenomenon.

Searle is clear that it is hard to see how to avoid these results on what he calls the "standard textbook definition" of computation. He is also quick to downplay his own point, envisioning an improved account of program implementation which requires the holding of certain causal relations and dispositions and presuming that his wall cannot meet them. Thus he is willing to see the absurdities as a consequence of a particular way of defining computation, one he attributes to Alan Turing. But some philosophers think that Searle’s concession here underestimates the power of his own objection.[3] And so we take Searle’s Wall to be a more serious challenge than he does; nevertheless, I think that it ultimately fails for a clear and systematic reason.

I will not delve into the issue of whether it is accurate to identify the source of these consequences as the "standard textbook view", nor will I argue against Searle’s result (1) or the first sentence of (2). In fact, I agree that it is hard to see how we can avoid these results, and I don't see a compelling reason to try. Additionally, I am prepared to accept that there is some pattern of molecule movements that is isomorphic to the formal structure of WordStar.[4] In accepting this much of Searle’s argument, I part ways with defenders of computationalism who argue either that mere isomorphism is insufficient for program implementation or that no such isomorphisms exist for any programs interesting enough to embarrass computationalism.[5] But I do think Searle errs in concluding from these premises that his wall implements WordStar along with “any program, including any program implemented in the brain.” In fact, I think it can be made rather easy to see how we can avoid this consequence and the subsequent generalization that if his wall is big enough it is implementing any program. I will attempt to do that here, defending computationalism and cognitivism in general, with two arguments.

2.

For heuristic reasons, I argue first by analogy. Galileo advanced the view that motion was relative: You may be at rest with respect to your bed but in motion with respect to the city, if your bed is onboard a passing boat. Motion holds relative to a frame of reference, and this is something we all accept. Imagine, however, some critic objecting as follows:

> On Galileo's definition of motion it is hard to see how to avoid the following results:
>
> 1. For any object there is some frame of reference according to which it is in motion.
>
> 2. For any velocity and for any object, there is some frame of reference according to which that object has that velocity. Thus for example the wall behind my back is right now traveling at 500 mph. But if the wall is traveling at 500 mph, then it is traveling at any velocity.[6]

No one today rejects point (1) or the first sentence of (2). But the subsequent attributions of velocities to the wall in (2) make it obvious that we must qualify these statements. The requisite qualifier is simply some acknowledgment that these attributions of traveling hold with respect to some frame of reference and not, we must understand, with respect to many others. These attributions of motion involve a three-place relation over an object, a velocity, and a frame of reference. Once this is acknowledged, the seemingly absurd claims can be dismissed as trivial.

A similar analogy can be made using the relativity of meaning.

> On the standard understanding of meaning it is hard to see how to avoid the following results:
>
> 1. For any object there is some interpretation according to which the object expresses a proposition.
>
> 2. For any proposition and for any object, there is some interpretation according to which it expresses that proposition. Thus for example the wall behind my back is right now expressing the proposition *that the unexamined*

> *life is not worth living*. But of course, it is also expressing the proposition *that the unexamined life is worth living* and infinitely many other propositions, many of them mutually contradictory.

The situation is much as before. These attributions of meaning involve a three-place relation: An object expresses a proposition only with respect to an interpretation.

Few philosophers these days would be impressed by these objections to our contemporary understandings of motion and meaning. Anyone that does is guilty of the following fallacies.

> If there is a frame of reference R according to which object x moves at velocity v, then x moves at v.
>
> If there is an interpretation I according to which object x expresses proposition p, then x expresses p.

In these cases, the fallacy is to mistake the relation in question for a mere two-place relation (expressed in the consequent) that is contingent on some other three-place relation (expressed in the antecedent). But, in fact, the relation in question is better understood simply to be the three-place relation.

Insisting on the conflated two-place relation, one merely contingent on a three-place relation, leads to absurdities. In the case of motion, I would have a pitching speed equivalent to that of Nolan Ryan. We all would. Worse, one could derive the result that right now, your left half travels at 50 mph forward while your right half travels at 50 mph backward. (And yet you live.) After all, there are such reference frames according to which each of these attributions holds. Moreover, in the case of meaning, zero information would be expressed. After all, 'Snow is white' would express the proposition *that snow is not white* as well as the propositions *that snow is white or not* and *that snow is white and it is not the case that snow white*. Of course, we easily manage to avoid these absurd consequences by avoiding the fallacies and the impoverished two-place conception that engenders them. We respect our reference frames and interpretive schemes.

An analogous fallacy is made when we accept Searle's argument that the standard definition of computation has the consequence that his wall implements WordStar or any other artificial program.

> If there is a description D under which object x implements program w, then x implements w.

The oversight has just not been as obvious. Accordingly, I propose that program implementation, like motion and meaning, is relative. It holds with respect not to reference frames or natural language interpretations but to descriptions of a certain kind: structure-preserving mappings from causally related physical states to programs. This is certainly not to say that the mere existence of such a mapping between some thing's causal structure and some program entails that the thing implements that program *tout court*. Implementation is not a two-place relation that is simply conditional on the existence of an appropriate mapping; instead it is a three-place relation where a thing implements a program according to a mapping.[7] And the three-place nature of implementation holds despite the fact that we often don't normally mention the mapping when we talk about implementation.[8]

Thus the proposed way of avoiding Searle's final consequences is to make explicit and retain as integral the dependence on a certain mapping of a certain kind. Specifically, my proposal is that we reject program implementation as a two-place relation that simply depends on the existence of some mapping or other, and instead embrace program implementation as a three-place relation over causal structures of physical things, programs, and mappings.[9]

This may not be an exciting proposal; in fact, it may look fairly trivial in hindsight. But I think it is also a faithful representation that is relevant to the problem of Searle's Wall. For now one cannot conclude simply that Searle’s wall implements WordStar just as one cannot conclude simply that the wall travels at the speed of light or expresses Shakespeare’s Sonnet 116. All we can conclude is that Searle’s wall implements WordStar according to some mapping M, that it travels at c with respect to some frame R, and that it expresses Sonnet 116 by some interpretation I. Whether M, R, and I are interesting or useful to us is of course another story.

So far, I have argued only by analogy that implementation is a three-place relation and that recognizing it as such allows one to reject Searle’s Wall. But there is an independent reason for acknowledging the implementation fallacy and embracing the analysis according to which implementation is a three-place relation. I present this in the next section.

3.

The implementation fallacy is inherently flawed, and thus off-limits to anyone who wants to speak coherently about computation, because it violates a basic constraint on program implementation:

> If a system transits from computational state A to computational state B at time *t*, then it does not transit from B to A at *t*.

This constraint follows from the notion of state transition and the very definitions of computational states A and B, whatever they may be. After all, if the system truly transits from state A to state B, then A and B must be non-identical. But if program implementation requires only the existence of some adequate mapping or other, then, incoherently, the system both transits from A to B at *t* and transits from B to A at *t*. Such a consequence is not bad news for program implementation because it cannot even be about program implementation. Program implementation requires that entered computational states be states entered to the exclusion of other states, but this view does not require it; in fact, it necessitates just the opposite. No states are occupied to the exclusion of any others, and therefore no systems ever leave any computational states. Specifically, Searle’s view not only results in his wall’s “implementing WordStar” as he sees it, but it also results in his computer’s being in every state of WordStar simultaneously. But then his computer thereby fails to implement WordStar, a program that, by its very definition and nature, requires nontrivial computational state change.[10] Implementation, then, construed as a two-place relation not only fails to establish that most objects implement most programs, but also fails to establish that any objects implement any programs at all.

On the other hand, if we insist on respecting the three-place nature of implementation, we can recognize that a system may transit from A to B with respect to mapping M at *t* but transit from B to A with respect to mapping M* at *t* as well. This is not unlike saying that I glide to my left with respect to reference frame R while simultaneously gliding to my right with respect to frame R*. In neither case are we in danger of being led to the conclusion that the system simultaneously occupies two states that were supposed to be mutually exclusive or that I simultaneously occupy two such locations.[11]

At this point a dialectical problem may arise, one that can arise when a philosopher points out, as I may seem to have, that a *reductio ad absurdum* fails because it yields more absurdity than originally thought. Why can’t Searle reply that, far from challenging his view, the foregoing just makes his point in another way? In exposing this further absurdity—the incoherence that results from the “standard textbook view” of computation—and in arguing for this third relatum, so the reasoning might go, I have only bolstered and clarified his thesis that computation is essentially observer relative. For, given that there are uncounted mappings, we are ultimately the ones who pick one mapping from among the others.

I agree that we indeed select mappings; however, I do not think this creates a problem for computationalism or cognitivism in general. We select mappings just as we select reference frames in physics. This makes mappings no more observer relative than reference frames.

4.

A question remains whether the three-place analysis of program implementation suits computationalism. After all, programs are supposed to be multiply realizable in great profusion and diversity. As Searle emphasizes, it is a commitment which computationalists “report with sheer and unconcealed delight” that programs can be realized by highly imaginative contraptions using pipes and water, tubes and marbles, even hens conditioned to peck on cue. Searle, however, holds that the relativity of computation appears to make matters even worse, taking us from multiple realization to virtually universal realization. As he sees it, no Rube Goldberg contraptions actually need to be built; we can just work out *ad hoc* mappings according to which “patterns of molecule movements” implement these programs. And yet, programs are also supposed by the computationalists to be the kinds of things we can discover to be implemented by systems such as brains through empirical research. What then ensures that the project of cognitive science is anything more than an opportunity for intellectual acts of *fiat* where, as long as some mapping can be defined by some bored theoretician, the brain can be “discovered” to implement any number of programs, WordStar included?

Searle summarizes his argument as follows:

> The aim of natural science is to discover and characterize features that are intrinsic to the natural world. By its own definitions of computation and cognition, there is no way that computational cognitive science could ever be a natural science, because computation is not an intrinsic feature of the world. It is assigned relative to observers. (p. 212)

In response, I will again resort to analogy. Imagine our skeptic of Galilean relativity advances the following objection.

> The aim of natural science is to discover and characterize features that are intrinsic to the natural world. By its own definitions of motion and location, there is no way that physics could ever be a natural science, because motion is not an intrinsic feature of the world. It is assigned relative to frames of reference.

Cognitive science maintains its scientific status despite the acknowledged relativity of computation in the same way that sciences as disparate as astronomy, ballistics, aeronautics, and plate-tectonics maintain their scientific status despite the fully acknowledged relativity of motion. But how, more exactly, does this work? In closing, I will urge that the scientific treatment of motion sets an example that cognitive science will surely benefit from following.

Note that significant attributions of motion are comparative and contrastive acts that reveal important features of the world. They are comparative because some things move like each other and some things don't. They are contrastive because a thing occupies one location or has one momentum to the exclusion of many other locations and momentums. We are usually interested in how stars move, projectiles fly, and continents drift with respect to other familiar things and locations, and these things and locations provide standard default reference frames for studying them. No geologist pretends to have discovered that Los Angeles is moving in the direction of San Francisco at a rate of 20 mph simply because there is a reference frame by which it is doing so. Instead, geologists, astronomers, and the others normally default to standard frames of reference, and they are careful to respect their reference frames when they use others. And while the two-place relations I have addressed here may be only observer (or frame, or interpretation, or mapping) contingent, the corresponding three-place relations either hold or fail to hold as an intrinsic and objective feature of this world, one that can be studied through direct empirical research.

There are three lessons here regarding motion, all familiar to a student of modern science. The first lesson is that attributions of motion, although relative to reference frames, must be made in ways that respect their reliance on those reference frames. The second lesson is that, for our purposes, some reference frames are better than others. The third lesson is that motion's relativity to reference frames is not itself a relative matter. The velocity of an object with respect to a reference frame holds absolutely, making it indicative of the intrinsic nature of systems.

These lessons can be learned, *mutatis mutandis* for program implementation. According to the first lesson, the mapping by which Searle's wall implements WordStar is certainly not the mapping by which his computer implements WordStar. And the mapping by which Searle's wall implements some more interesting cognitive program is certainly not the mapping by which brains implement it. According to the second lesson, these mappings, while they are just as real as any others, are of no use to our empirical research. The mappings that may be useful are any according to which certain personal computers run WordStar, while walls do not, and brains run elaborate cognitive programs, while walls and contemporary personal computers do not. Finally, according to the third lesson, program implementation now construed as a three-place relation is indeed a feature of things in the world that holds absolutely, making it indicative of the intrinsic nature of things.

Significant attributions of program implementation, like those of motion and meaning, are comparative and contrastive acts. What we seek empirically is some default standard mapping, that mapping by which we can make objective discoveries of relevant similarities and differences holding not only among the brains of various biological organisms but also among artificial systems. Only by such a mapping will it be significant to show that an unusual or artificial system implements some or all of the cognitive programs certain brains implement by that very same mapping. The computationalist believes that some of these systems may be in most physical aspects nothing like a brain; they may be comparatively bizarre in substance or structure, alien or inorganic. But none of these systems, if we ever succeed in discovering or creating them, should be anything like Searle’s wall.

[1] Hilary Putnam’s argument that “every ordinary open system is a realization of every abstract finite automaton” (1988, appendix) has been treated by philosophers as making the same case. (Chalmers 1996, Scheutz 1999, 2002, Bishop 2002, Rey 2002)

[2] Searle’s charge here is thus much more radical than that made by his Chinese Room thought experiment or by Ned Block’s Chinese Nation thought experiment. (Searle 1980, Block 1978) In these cases, program implementation is seen to require severe physical regimentation either of an inner rule-following human or by an entire population of humans. Searle’s wall, however, can be left alone. All that is required is that there be some way of interpreting whatever it happens to be doing that is isomorphic to the formal structure of some desired program.

[3] Many philosophers (Chalmers 1996, Copeland 1996, Block 2002, Haugeland 2002, Rey 2002) have objected that a thing such as Searle’s wall cannot have the behavioral dispositions that would meet counterfactual constraints imposed by some programs. But, intuitive as such appeals may be, they face two problems. First, some authors claim that it doesn’t always make sense to require counterfactual constraints. (Maudlin 1998, Scheutz 1998, Bishop 2002) Second, it has not been shown that the right counterfactuals cannot be secured; in fact, the means (however dubious) by which one can argue that a sufficiently complex wall displays the requisite actual behavior can be generalized to argue that the wall is disposed to display the requisite counterfactual behavior. The idea here is that whatever descriptive liberties apply to the actual physical state changes of the wall can be applied to counterfactual state changes as well, putting these enhanced counterfactual-supporting descriptions on equal footing with the original inspirations we find in Putnam (see footnote 1) and Searle. David Chalmers (1996) exploits this opportunity in order to enhance Putnam’s argument, adding to the physical system a dial which has, for each other way the program could have been run, a position the dial could have been in. Paul Teller, in an unpublished draft (1999), exploits the unbounded potential to use refined physical state descriptions (also exploited by Putnam) to secure not only the requisite counterfactuals, but input-output relations and a kind of internal complexity. The foregoing gives some indication why I think Searle underestimates the power of his objection. For the purpose of this paper, which is to identify the real problem with Searle’s Wall, I grant that it can be shown that Searle’s wall is so disposed.

[4] Isomorphism is widely acknowledged to be cheap. For some arguments establishing isomorphism (albeit in these cases between a variety of significantly different kinds of physical and computational entities), see Putnam (1988), Chalmers (1994, 1996), Copeland (1996), and Scheutz (1998, 1999). These arguments may require using seemingly *ad hoc* definitions in order to acknowledge odd, disjunctive, or

gerrymandered physical states, but as I think a survey of this literature shows, justifying their rejection has proven difficult and controversial.

[5] See Chrisley (1994), Copeland (1996), Haugeland (2002), and Rey (2002) for instances of the former and Chalmers (1994, 1996) and Block (2002) for instances of the latter.

[6] Although this may not have bothered Galileo, speeds greater than the speed of light are included.

[7] Searle frequently insists that computational features are not intrinsic features of the world but are instead observer relative ones. This may leave the impression that he and I agree at least on the three-place nature of the implementation relation, disagreeing only on whether the third relatum is an observer, as he insists, or a description, as I propose. But this is to overlook the important distinction just made, that distinction between a two-place relation that is merely conditional on a third thing and a three-place relation that includes that third thing as a relatum. In light of this distinction, it is more accurate to say that Searle has been treating program implementation as being merely *observer contingent*, as if it holds whenever an observer can see it as holding. But this cannot be right, as I think my second argument, presented in the next section, will show.

[8] By the same token, the fact that we rarely cite frames of reference or linguistic interpretations is not taken to imply that motion or meaning does not hold in ways that are relative to these things.

[9] Mappings have been specified in a variety of ways. Copeland (1996), for example, takes a model theoretic approach, while Putnam (1988) stipulates "definitions" of computational states in terms physical states. But however a mapping is specified, it must at least be equivalent to a function from physical states of an object to computational states of a program such that the causal relations among physical states preserve the transition relations among computational states. What exactly counts as a physical state or a computational state I leave open. (Putnam (1988), Chalmers (1996), and Copeland (1996) are willing to consider grossly disjunctive physical states.) Moreover, what counts as an appropriate causal relation is left open. The strength of my point that implementation is a three-place relation over causal structures, programs, and mappings does not rely on the specific nature of mappings.

[10] Alan Turing's original exposition (1936) of computing machines makes it clear that states are mutually exclusive and that state change is thus nontrivial, and they are routinely treated as such by practitioners.

[11] Similar remarks hold for meaning.

Acknowledgements

Special thanks to two anonymous referees for this journal who made helpful comments and to Andrew Melnyk for helpful comments on a previous draft presented at the 2011 APA Conference, Central Division. I am also very grateful to Paul Teller, Robert Cummins, Jonathan Dorsey, John Keller, and Michael Jubien for valuable discussions and commentary regarding earlier drafts of this paper.

References

Bishop, M. (2002). ‘Dancing with Pixies’. (In Preston and Bishop (Eds.) 2002.

Block, N. (1978). ‘Troubles with Functionalism’, in C. W. Savage (Ed.), *Perception and Cognition: Issues in the Foundations of Psychology*, Minneapolis: University of Minnesota Press.

Block, N. (2002). ‘Searle's Arguments Against Cognitive Science’, in Preston and Bishop (Eds.) 2002.

Chalmers, D. J. (1994). ‘On implementing a computation’. *Minds and Machines*. 4:391-402.

Chalmers, D. J. (1996). ‘Does a rock implement every finite-state automaton?’. *Synthese* 108:309-33.

Chrisley, Ron, (1994). ‘Why everything doesn’t realize every computation’. *Minds and Machines*. 4:403-20.

Copeland, B. J. (1996). ‘What is computation?’. Synthese. 108, 355-59.

Haugeland, J., (2002). ‘Syntax, Semantics, Physics’, in Preston and Bishop (Eds.) 2002, 379–392.

Maudlin, T. (1989). ‘Computation and consciousness’. *The Journal of Philosophy*, vol. LXXXVI, no. 8, pp. 407-432.

Preston, J. and M. Bishop (Eds.) (2002). *Views into the Chinese Room: New Essays on Searle and Artificial Intelligence*, New York: Oxford University Press.

Putnam, H. (1988) *Representation and Reality*. MIT Press.

Rey, G. (2002). ‘Searle's Misunderstandings of Functionalism and Strong AI’, in Preston and Bishop (Eds.) 2002, 201–225.

Scheutz, M. (1998). ‘Implementation: Computationalism’s Weak Spot’. Conceptus JG, 31, 79, 229-239.

Scheutz, M. (1999). ‘When Physical Systems Realize Functions’. Minds and Machines. 9:161-196.

Searle, J. (1980). ‘Minds, Brains and Programs’, *Behavioral and Brain Sciences*, 3: 417–57.

Searle, J. R. (1992). *The Rediscovery of the Mind*. MIT Press.

Teller, P., (1999). ‘Rocks can think’, unpublished draft.

Turing, A. (1936). ‘On Computable Numbers, With an Application to the Entscheidungsproblem’, *Proceedings of the London Mathematical Society*, 42 (1936), pp. 230-265.