The Problem of Pancomputationalism: Focusing on Three Related Arguments[*]

SeongSoo Park
Department of Philosophy, Sungkyunkwan University
seongsoo@buffalo.edu

Abstract

Pancomputationalism is the view that everything is a computer. This, if true, poses some difficulties to the computational theory of cognition. In particular, the strongest version of it suggested by John Searle seems enough to trivialize computational cognitivists' core idea on which our cognitive system is a computing system. The aim of this paper is to argue against Searle's pancomputationalism. To achieve this, I will draw a line between realized computers and unrealized computers. Through this distinction, I expect that it will become evident that Searle's pancomputationalism should be understood in terms of unrealized computers, while the computational theory of cognition is concerned with realized computers.

Keywords: *Pancomputationalism, Computational Theory of Cognition, John Searle, The Chinese Room Argument, Functionalism*

1. Introduction

The computational theory of cognition (or CTC for short) propounds that

Manuscript received 8 January 2019; Revised version received 1 June 2020; Accepted 13 May 2019
Journal of Cognitive Science 21-2:349-369, 2020
©2020 Institute for Cognitive Science, Seoul National University
* I am grateful to David Braun for motivating me to write this paper. Thanks to Peter Baumann for his many pertinent discussions of an earlier draft of this paper. Finally, thanks to the anonymous referees for the Journal of Cognitive Science for very useful comments. This work was supported by the grant from the National Research Foundation of Korea (NRF-2019H1A2A1075330).

cognition is computation. On this view, a cognitive system can be explained in terms of the computations it performs, because cognitive processes are computational. Therefore, proponents of this view (e.g., McCulloch & Pitts, 1943; Newell & Simon, 1976; Pylyshyn, 1985; Edelman, 2008; Schneider, 2011) claim that the human cognitive system implements (or executes) cognitive programs that allow us to think and make decisions.

One of the most pertinent challenges to CTC is the so-called 'strongest version of pancomputationalism' suggested by John Searle.[1] In his 1990 paper, "Is the Brain a Digital Computer?" Searle argues that everything is a computing system that performs every computation. According to him, even a wall is a computational object that can perform every program. This claim, if true, raises a hard problem for CTC because it trivializes CTC.

In this paper, I argue in favor of CTC, addressing three related arguments regarding pancomputationalism. Specifically, I argue that although the Chinese Room Argument is convincing, there are good reasons not to accept the Pan Argument. To illustrate this, I will draw a line between realized computers and unrealized computers. Through this distinction, I expect that it will become evident that Searle's pancomputationalism, if properly understood, does not provide any adequate reason to deny CTC since it should be understood in terms of unrealized computers, rather than realized computers.

2. The Triviality Argument

Searle (1990) attempts to argue against CTC by endorsing the strongest version of pancomputationalism, according to which everything computes any program. However, someone might say, "So, why does pancomputationalism matter to computational cognitivists?" This seems to be a natural reaction in that almost all computational cognitivists (e.g.,

1 Putnam also made a similar claim in his 1988 paper.

Smart, 1959; Fodor, 1968; Newall & Simon, 1976; Pylyshyn, 1984; Horgan & Woodward, 1985) accept the multiple realizability thesis of computing objects, which propounds that there can be various types of computers that are comprised of different physical materials, since they gladly admit that computation can be functionally defined.

Despite its seemingly favorable appearance, the claim that everything is a kind of computer that can implement (or execute) any program, if true, poses several serious problems for CTC. One problem is that it plays a crucial role within the *Triviality Argument.*[2] The argument can be formally represented as follows:

1. The computational theory of cognition states that our cognitive system is a computing system.
2. Everything is a computing system that performs any computation.
3. If (1) and (2) are true, then the computational theory of cognition is not informative.
4. Therefore, the computational theory of cognition is not informative.

Please note that the aim of the Triviality Argument is not to refute CTC. Rather, the argument aims to illustrate that CTC says little about how our cognitive system works or how it produces mental phenomena. In other words, the key point of the Triviality Argument is that if the strongest version of pancomputationalism is true, then there is nothing special in the claim that human cognitive processes are computational.

Before we go through the argument, it is also worth noting that numerous versions of pancomputationalism exist. Not all versions of pancomputationalism make CTC trivial. For instance, a weaker one, which

2 Although Searle did not suggest the Triviality Argument explicitly, it is commonly held that Searle's challenge to CTC can be understood in this way. See Piccinini 2010 and Rescorla 2015.

propounds that everything is a computing system that implements some computation, does not harm CTC. This is because CTC may secure its own explanatory power by arguing that our cognitive system implements a unique program that other computers are not able to execute.[3]

It is my belief that the only way to refute the Triviality Argument is to deny the strongest version of pancomputationalism. First, all proponents of CTC must accept premise (1), since what premise (1) indicates is the core thesis of the computational theory of cognition. Second, premise (3) is true. Assume that premises (1) and (2) are true. This implies, as Piccinini (2010) puts it, that CTC is trivial and vacuous since the human cognitive system is already accounted for by pancomputationalism. Searle (1990) states, (if one asks whether our cognitive system is a computer) "it would not answer that question to be told that the brain [or human cognitive system] is a digital computer in the sense in which [the] stomach, liver, heart, solar system, and the state of Kansas are all digital computers" (p. 26). In other words, given that everything is a computer that performs every computation, the claim that the human cognitive system executes cognitive programs is not interesting at all.

Thus, the remaining option for proponents of CTC is to deny premise (2). That is, computational cognitivists should deny the strongest version of pancomputationalism. The way I see it, this option is promising despite the efforts Searle made in his 1990 paper.

3. The Chinese Room Argument

Before presenting Searle's argument for pancomputationalism, I will first address another argument related to pancomputationalism, the *Chinese*

3 For example, Bahara and Piccinini (2013) argue that neural computation is sui generis. Thus, according to them, there is a certain kind of program which is only implemented by neural processes.

Room Argument. The reason is that the Chinese Room Argument, if true, supports the core premise of the argument for pancomputationalism which states that computation is defined exclusively in terms of the assignment of syntax.[4] This premise is crucial in that it elucidates the nature of computation.

The Chinese Room Argument was originally suggested by Searle to deny the position he called *Strong AI.* The position can be rearticulated as follows:

> Strong AI: for any mental property, there is a program such that anything that can run/execute the program, and executes the right steps in that program, has that mental property.

To comprehend this point, let us consider AlphaGo, a deep-learning program that was designed to beat the world Go champion. The program indeed managed to beat Sedol Lee, a high-ranking Go player in 2016. The point is that if a proponent of Strong AI had watched the surprising match between AlphaGo and Sedol Lee, she would have argued that AlphaGo *understands* how to play Go. This is because, according to Strong AI, running a program is not distinct from having a relevant mental property. In this view, for example, anything that implements a pain-program must be in a pain state. This is the key point of Strong AI.

The Chinese Room Argument claims that AlphaGo does not understand how to play Go. To illustrate this, Searle (1999) asks us to consider the

4 As an anonymous referee points out, the claim that computation is defined exclusively in terms of the assignment of syntax is a general claim, so it might be supported by other arguments than the Chinese Room Argument. Still, I think the Chinese Room Argument is relevant to the issue of pancomputationalism. This is because one could stick to refuting it to weaken the argument for pancomputationalism. Also, as a matter of fact, Searle developed his view on pancomputationalism from the Chinese Room Argument. Thus, addressing the Chinese Room Argument will help us see the line of thoughts Searle had.

following thought experiment:

> Imagine a native English speaker who knows no Chinese locked in a room full of boxes of Chinese symbols (a data base) together with a book of instructions for manipulating the symbols (the program). Imagine that people outside the room send in other Chinese symbols that, unknown to the person in the room, are questions in Chinese (the input). And imagine that by following the instructions in the program, the man in the room is able to pass out Chinese symbols which are correct answers to the questions (the output) (p. 115).

After suggesting the above scenario, Searle (1999) asks, "Does the man in the room understand Chinese?" (p. 115). The answer seemingly is in the negative. It appears that the man in the room may perform the given task without understanding any Chinese. The problem is that if Strong AI is true, the answer must be in the affirmative. For this reason, Searle argues that Strong AI is false. Now let me call the program described in the above scenario CU. Then, the argument can be formally represented as follows:

1. The man in the room runs the CU program.
2. If (1), then, if Strong AI is true, the man in the room understands Chinese.
3. Thus, if Strong AI is true, then the man in the room understands Chinese.
4. It is not the case that the man in the room understands Chinese.
5. Therefore, Strong AI is not true.

On the one hand, as the argument indicates, the Chinese Room Argument can be construed as an attempt to prove that Strong AI is false. However, in a broad sense, the argument can also be construed as an attempt to prove that computation should be defined exclusively in terms of symbol manipulation. Searle (1990) notes that "a system, for example, the man in the room, would not acquire an understanding of Chinese just by going through the steps of a computer program that simulated the behavior of a Chinese speaker" (p. 17). If so, given that the man in the room cannot derive any meaning from the formal symbols, it seems that we should consider computational processes to be purely syntactical processes. As we will see in the next section, the argument for pancomputationalism begins with this claim. In this light, the Chinese Room Argument is directly related to Searle's argument for pancomputationalism. Therefore, one might attempt to argue against the Chinese Room Argument to deny Searle's pancomputationalism.

However, I think that the Chinese Room Argument is persuasive, despite its criticisms. The best-known criticism is the System Reply, suggested by Georges Rey (1986) and Rey Kurzweil (2002). The basic strategy of the System Reply is to deny the Chinese Room Argument's first premise, which states the man in the room is running the CU Program. According to Rey and Kurzweil, the man in the room is just a CPU, not a system. Thus, the man in the room is not running the CU program alone. Rather, the entire room, which contains the boxes of Chinese symbols, the book of instructions, and the man, is running the CU program. This implies that, if the System Reply is correct, the man in the room does not need to understand Chinese. The entity that is supposed to understand Chinese is the entire system.

Unfortunately, the claim that the entire system understands Chinese is also false. This can be argued as follows. Suppose that the man in the room memorizes all the Chinese symbols and also all the instructions for manipulating the symbols. Then, the man will be the entire system. Still, the man does not appear to actually understand Chinese. He does not know what the Chinese characters mean. He can perform his task without knowing English either as long as the instructions are written in his native language. Thus, I conclude that the System Reply is not adequately persuasive.

Other critics, such as Rey and Fodor argue that the thought experiment Searle suggested is not enough to prove that Strong AI is false. They admit that a computer fixed within the Chinese Room cannot understand Chinese. However, they argue that a moving robot implementing the CU program with video cameras and microphones can understand Chinese by seeing, hearing, and using Chinese characters. In other words, Rey and Fodor believe that a computing system may be associated with meaning if we allow it to have additional inputs related to sensations.

However, this sort of criticism also seems implausible. No matter how many inputs the robot has, the system cannot derive any meaning. The man who memorizes all the instructions and Chinese symbols, of course, has genuine sensations. However, we concluded that the man does not understand Chinese. Correspondingly, a moving robot with video cameras and microphones also does not understand Chinese.

4. The Pan Argument

Is everything a computing system that implements (or executes) any program? According to Searle (1990), the answer is "yes." This is an astounding claim, because by its logic, a situation wherein a rock implements the Facebook program is presumed to be legitimate. Indeed, Searle argues that the wall behind him is a computer that implements the WordStar Program.[5] At first glance, this claim seems intuitively false, but Searle attempts to influence our intuition by suggesting a seemingly plausible argument. I will call this argument the *Pan Argument.*

The upshot of the Pan Argument is that the implication of the Chinese

5 The WordStar program is an old-fashioned typing program.

Room Argument that computation should be defined exclusively in terms of symbol manipulation leads us to the devastating conclusion that everything is a computer implementing any program. Indeed, Searle (1990) says, “if computation is defined in terms of the assignment of syntax then everything would be a digital computer” (p. 26).

It is rather unclear, *at this stage,* how we get to reach this astounding conclusion from the syntactical definition of computation. So, let me first formally present the Pan Argument that Searle devised, and then explain it in detail as follows:

1. Computation is exclusively a matter of syntax.
2. If (1), then being a computer is not a matter of physics.
3. Therefore, being a computer is not a matter of physics.
4. If (3), then computation is observer-relative in the sense that assigning syntactical properties is always relative to an agent.
5. Therefore, computation is observer-relative in the sense that assigning syntactical properties is always relative to an agent.
6. If (5), then everything is a computing system that performs every computation.
7. Therefore, everything is a computing system that performs every computation.

As can be observed, line (1) comes from the syntactic conception of computation. Computational processes are algorithmic. A computing system can execute a given algorithm by simply manipulating the binary symbols—0 and 1. Therefore, computation is defined purely in terms of the assignment of 0s and 1s and the way they are transformed. To implement a program, all that a computational object needs to do is simply manipulate 0s and 1s according to the set of instructions built in the program. Thus, it follows that computation is syntactic: computer languages, like 0s and 1s, are properly arranged by the set of language rules specified by a program.

The syntactic conception of computation has a crucial implication. It seems to entail, as in line (2), the universal realizability of computers—meaning a computer can be made of almost anything. If we have some physical materials, which could be treated as 0s and 1s, and some instruction (or description) that specifies a set of rules regarding symbol manipulations and the assignment of 0s and 1s, we can use them to create a computational object. A set of cups serves as a good example. Suppose there is a set of cups in front of us. Now suppose that we treat the cups filled with water as 1s and treat the empty cups as 0s. If we have instructions that specify the rules of a simple addition program, then, by using the cups, we can make a computer that can perform the addition, albeit in an extremely inefficient way. In a similar vein, "a computer can even be made out of a group of pigeons trained to peck" (Pylyshn, 1985, p. 57). This universal realizability indicates that symbols and a set of rules regarding them are not intrinsically physical and hence the computation defined by them is not either. In other words, we can never find syntax and symbols in our physical world. They are just assigned to objects by us. As a result, physical materiality is irrelevant to the conceptualization of a computer.

Searle followingly argues that accepting the claim that computation is not intrinsic to physics is tantamount to accepting the claim that computation is observer-relative. In other words, according to Searle (1990), the fact that syntax and symbols are just assigned to an object means that "assigning syntactical properties is always relative to an agent who treats certain physical phenomena as syntactical" (p. 26). It is worth noting that Searle uses the notion of observer relativity here in an extremely strong sense. He says, "you can assign any computational interpretation to anything" (Searle, 2002, p. 17). That is, syntactical properties, sometimes also called computational interpretations, such as assigning a set of rules or treating something as 0s or 1s, can be given arbitrarily to any physical object. Searle thus concludes that computation defined solely in terms of the assignment of syntax is observer-relative.

This argument seems cogent thus far. However, Searle takes it a step further by claiming that it follows from (5) that the wall behind him is a computer that is performing the WordStar program. He writes:

> For any program there is some sufficiently complex object such that there is some description (or some instruction) 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 which is isomorphic with the formal structure of WordStar. (Searle, 1990, p. 27)

The thought underlying this passage is simple. If assigning syntactic properties are relative to an agent, then we can regard the wall as a computer that is implementing the WordStar program, since we can surely assign the same syntactic properties as the WordStar program to some pattern of the wall's molecular movements.

If the wall behind Searle, as he claims, is a computer that is implementing the WordStar program, then it can be argued that everything is a computer that implements every program. However, is the wall really a computer? It certainly does not appear to be. Intuitively, there seems to be something misleading with the Pan Argument, and this opinion is shared by many.

5. Criticisms of the Strongest Version of Pancomputationalism

Some critics believe that premise (4) of the Pan Argument is dubious. For instance, Georges Rey (2002) argues that observer relativity does not follow from the claim that computation is not a matter of physics. Specifically, he insists that the fact that certain categories are not definable to physics does not imply that *that* category is observer-relative. To consolidate his argument, Rey mentions micro-science as an example. He says, “Many categories of the micro-sciences (being a species, being an eco-system) may not be definable to physics without there being the slightest reason to think they are observer relative” (Rey, 2002, p. 215).

However, in my opinion, this criticism is not persuasive. This is because premise (4) is not a general claim—that is, premise (4) is not about other categories, like those in micro sciences, but just about computation. Probably, the reason some micro-sciences are not observer-relative even though they may not be definable to physics is that the relevant concepts used in micro sciences are not purely syntactical.

Of course, this does not demonstrate that premise (4) is true either. Therefore, one could still attempt to deny this premise. One of Ned Block’s points aim to do this. Block (2002) argues that there are two constraints to implementing computation. He states that (1) there is a certain logical limitation that prevents us from assigning syntactic properties in an arbitrary way, and that (2) a computing system must have the capacity to perform counterfactual computations that have not been implemented.

With regard to the first point, Block (2002) asks us to consider the logical forms of logical connectives. For example, according to him, it is possible for us to convert any physical device that can be interpreted as an AND gate into a physical device that can be interpreted as an inclusive OR gate by replacing its existing interpretation with the opposite one.[6] In

6 Two notes: First, roughly, an AND gate is a physical device that implements logical conjunction. It behaves according to the truth table of “and.” So, an AND gate operates according to the following set of rules: (0,0) = 0, (0,1) = 0, (1,0) = 1, (1,1) = 1 (where ‘1’ stands for true, and ‘0’ stands for false).
Second, an OR gate is a physical device that implements logical disjunction. It behaves according to the truth table of “or.” There are two types of the OR gate— the inclusive OR gate and the exclusive OR gate—which depend on how the meaning of “or” is interpreted. So, an inclusive OR gate operates according

other words, if we replace 1s with 0s and replace 0s with 1s, we can obtain a physical device that can be interpreted as an inclusive OR gate from a physical device that can be interpreted as an AND gate. However, despite this fact, it is still not possible for us to convert the logical structure of an inclusive OR gate into an exclusive OR gate. This is because their inherent logical structures are different. Thus, according to Block, the computation is not fully observer-dependent.

However, in my opinion, this objection is not satisfactory either. The reason is that Searle may argue that the logical structure of an inclusive OR gate could be broken down to a lower physical level just as he thought of a wall as a pattern of molecule movements. If so, the inclusive OR gate may have a broader logical structure than the one it originally has, which may allow for its conversion into an exclusive OR gate.

I think Block's second objection, which denies premise (6), is more persuasive. Specifically, Block points out that Searle's wall is not a computing system because it has no capacity to perform all the possible state transitions that are counterfactually described by the program. Block (2002) argues, "The wall has to include not just a particular computation that the machine does perform, but all the computations that the machine could have performed" (p. 77). To illustrate this point, let us assume that I type the word 'apple' with the WordStar program. It is obvious that I could have typed other words, such as 'mushroom,' instead of 'apple.' Block's point is that the wall could have never implemented the above possible command because the wall had already been assigned specific computational interpretations. As a result, the observer-relativity of computation does not guarantee the plausibility of the claim that the wall is a computer, which leads to the refutation of premise (6).

to the following set of rules: (1,1) = 1, (1,0) = 1, (0,1) = 1, (0,0) = 0. Whereas an exclusive OR gate behaves according to the following set of rules: (1,1) = 0, (1,0) = 1, (0,1) = 1, (0,0) = 0.

Still, one might argue in favor of Searle by stating that I could have interpreted the wall in a different way after dropping the first interpretation. By this logic, one might claim that the wall could have implemented the command of typing 'mushroom' by virtue of being assigned the second interpretation.

I am not convinced by this reply. The reason for this seems straightforward. Manipulating the existing symbols in a different way should not be conflated with interpreting the computational object in a different way. The point here is not whether we can interpret a computing system differently. Rather, the point is whether we could have implemented other counterfactual computations by manipulating the already assigned symbols differently. From my perspective, this constraint seems fatal to Searle's pancomputationalism. Ordinary computers implementing the WordStar program could have performed another task by going through other state transitions upon the provision of different inputs. However, it is obvious that the wall in question cannot function in this way.

Interestingly, unlike the wall, the computing system made of cups that was mentioned in the previous section could have implemented another task if a user wanted to do so. Presumably, the difference lies in whether the objects in question that are under a computational interpretation can be manipulated. The cups treated as 0s and 1s can be manipulated, and hence they could have been manipulated in a different way. On the contrary, in the case of the wall, there is no way for us to manipulate the pattern of its molecular movements. This means that if an object is a kind of computer, the object must be able to, whether internally or externally, be manipulated. Though Block does not explicitly mention it, I argue that this is the core constraint of being a computer that we can bring out from his second objection. As a result, I conclude that everything is not a computer because only the sort of thing that can be manipulated *can be* a computer. This point will be further explored in the following section.

6. Realized Computers Versus Unrealized Computers

My own criticism of Searle's pancomputationalism is that Searle does not have a fine-grained conception of a computer. The way I see it, his coarse conception of a computer misleads him to believe that his wall is a computer. In this section, I will provide an additional reason to deny the claim that the wall is a computer by drawing a line between realized computers and unrealized computers.

To begin with, I concede to Searle's point that there is a description under which an object is implementing the WordStar program. However, I believe that the pattern of molecular movements of the wall behind Searle is *not* under such a description. This is because Searle did not assign any specific type of computational interpretation to the pattern. It seems clear to me that Searle did not. He did not even mention how the assignment of 0s and 1s should be arranged to implement the WordStar program. The only thing he mentioned is that it was a possibility.

This point would be more evident if we adopt a more fine-grained conception of a computer than Searle. Specifically, I distinguish an unrealized computer from a realized computer.[7] An unrealized computer is a physical object (or a collection of physical objects) that merely has the possibility to be realized as a computer. This possibility is realized only when an appropriate computational interpretation is provided. For instance, let us assume that there is a set of cups before us. It seems obvious that those cups are not a real computer in themselves. They are just a bunch of cups until we assign a specific computational interpretation to them.

If we accept this distinction, it becomes evident that Searle's treatment of pancomputationalism in terms of realized computers is false. Pancomputationalism can at most be understood in terms of unrealized

7 In this paper, I use the term 'realized computer' to refer to a concrete computer that has already been assigned a specific description.

computers. In fact, the view, if properly understood, is only committed to a modal claim. At most, pancomputationalism can propound that everything is an unrealized computer that might be realized in the future by virtue of receiving some specific computational interpretation. The wall is not a realized computer of the kind we can buy in the store in the sense that it has not been assigned any specific type of computational interpretation. If so, the original Triviality Argument should also be understood in a modal way as follows. I will call this modified version of it the *Modal Triviality Argument.*

1. The computational theory of cognition states that our cognitive system is a computing system.
2. Everything *could be* a computing system that performs any computation.
3. If (1) and (2), then the computational theory of cognition is not informative.
4. Therefore, the computational theory of cognition is not informative.

I do not think that this is the result that Searle wanted to argue for. Above all, CTC is not refuted by this argument. As we can see in premise (2), the Modal Triviality Argument adopts a modal understanding of pancomputationalism. According to this understanding, everything is not a computer performing any program, but everything could be a computing system that performs any computation. However, this modal understanding of pancomputationalism is also false, since not all physical objects are unrealized computers. This can be argued as follows:

First, some objects (or collections) might be too simple to be an unrealized computer. Searle believes his wall is a computer that implements the WordStar program because some pattern of the wall's molecular movements is sufficiently complex for the isomorphic formal structure of the WordStar program to be assigned to it. This implies that a complex computational interpretation, like cognitive programs, may not be assigned to a simple object.[8]

Second, following the implication of Block's second objection, I argue that only the sort of physical object (or collection) that can be syntactically manipulated is an unrealized computer. If an object cannot be syntactically manipulated, then the object has no possibility to be realized as a computer, even if a certain computational interpretation is provided to it. This is because it lacks the capacity to perform all the possible state transitions that could be required. Hence, I conclude that the modal understanding of pancomputationalism is false, which means that the Modal Triviality Argument is unsound.

In what follows, I will briefly consider the last criticism an opponent of CTC might attempt to levy. To do this, let us finally consider the following version of the Triviality Argument, which might be called the *Restricted Modal Triviality Argument.* (Where type X indicates the type under which an object can be regarded as an unrealized computer. That is, type X refers to unrealized computers.)

1. The computational theory of cognition states that our cognitive system is a computing system.
2. Every object of type X (i.e., every unrealized computer) could be a computing system that performs any computation.
3. If (1) and (2), then the computational theory of cognition is not informative.
4. Therefore, the computational theory of cognition is not informative.

8 Nevertheless, the condition of being sufficiently complex is a modest condition in that almost all physical objects satisfy this condition. Suppose that there is a program such that it has only the following rule: yield 1 as the result when 0 is given. Then, having a physical ground that is enough to be assigned this description is not that hard.

This argument is clearly not the one that Searle has in mind, but one might think that the Restricted Modal Triviality Argument provides a good reason to refute CTC. However, this argument is also unsound. Above all, Premise (2), like the previous case, adopts a variant of pancomputationalism, which is still based on a modal understanding. The problem is that the notion of a computer that CTC adopts refers to a realized computer, not an unrealized computer. As Michael Rescorla (2015) puts it, nearly all proponents of the computational theory of cognition accept the claim that Mentalese is realized by neural states on the basis of the language of thought hypothesis, and computational operations over Mentalese symbols are realized by neural processes. According to computational cognitivists, the behaviors of our cognitive system—that is, computations—occur over Mentalese. Just like a concrete computer manipulates symbols, our cognitive system operates over language-like in the mind. If so, the computational theory of cognition should be understood in terms of a realized computer.[9] However, premise (2) of the Restricted Modal Triviality Argument does not relate to the notion of a realized computer. Hence, the argument does not refute CTC.

Moreover, premise (2) is also false. For example, let us suppose that some unrealized computers are only complex enough to perform an addition program. If so, they cannot be a computer that implements a complex program. That is, they are complex enough to be computers, but they are still too simple to implement certain kinds of programs. This implies that only some physical objects within type X could qualify as computers that can perform cognitive programs. The point is that CTC is not refuted by the fact that some objects within type X could potentially be realized as computers that run cognitive programs. Indeed, this is the claim that

9 The claim that our cognitive system is a realized computer can be made in different ways. Piccinini is one of philosophers who discusses this. See Piccinini 2012.

proponents of CTC want to highlight. According to computational theorists of cognition, our cognitive system, in its nature, is not that different from sufficiently complex computers. This is why they attempt to explain our cognitive system in terms of computation. Therefore, as long as it does not trivialize CTC, computational theorists of cognition can embrace the fact that some physical objects within type X could potentially be realized as computers that can implement cognitive programs.

7. Conclusion

This completes my defense of the computational theory of cognition. I conclude that the Triviality Argument is unsound because the strongest version of pancomputationalism is false. In showing this, I did not directly engage with the Chinese Room Argument. I have nothing to say on the syntactic conception of computation. However, I have many words to say about the Pan Argument. The observer-relativity of computation does not guarantee that everything is a realized computer. Any object cannot become a computer before a specific computational interpretation is actually given to it. Hence, the strongest version of pancomputationalism is false. Furthermore, a modal understanding of it is also unconvincing, since not all physical objects are unrealized computers. Thus, the most legitimate move of pancomputationalism is merely that some physical objects belonging to the category of unrealized computer could be a computer that performs any program. This peculiar version does not threaten the computational theory of cognition. Rather, this is the core claim that computational cognitivists, if they adopt the syntactic conception of computation, should embrace.

References

Blackmon, J. 2013. Searle's Wall. *Erkenntnis*, 78, 109-117.

Block, N. 2002. Searle's Arguments against Cognitive Science. In J. Preston and M. Bishop (Eds.), *Views into the Chinese Room*, 70-79. Oxford University Press.

Chalmers, D. 1994. On Implementing a Computation. *Mind and Machines*, 4, 391-402.

Edelman, S. 2008. *Computing the mind: How the mind really works.* Oxford: Oxford University Press.

Fodor, J. 1968. *Psychological Explanation.* New York: Random House.

Horgan, T., & Woodward, J. 1985. Folk Psychology is Here to Stay. *Philosophical Review,* 94, 197-226.

Kurzweil, R. 2002, Locked in his Chinese Room. In J. W. Richards. (Ed.), *Are We Spiritual Machines: Ray Kurzweil vs. the Critics of Strong AI*, 128-171. Seattle: Discovery Institution.

McCulloch, W. S., & Pitts, W. H. 1943. A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*, 7, 115-133.

Newell, A., & Simon, H. A. 1976. Computer science as an empirical enquiry: Symbols and search. *Communications of the Association for Computing Machinery*, 19, 113-126.

Piccinini, G. 2010. Computation in Physical Systems. *Stanford Encyclopedia of Philosophy.*

Putnam, H. 1960. Minds and Machines. In S. Hook (Ed.), *Dimensions of Mind: A Symposium*, 138-164. New York: Collier; Reprinted in Putnam 1975, 362–386.

Putnam, H. 1980. Models and Reality. *Journal of Symbolic Logic*, 45, 464-482.

Pylyshyn, Z. 1985. *Computation and Cognition.* Cambridge, MA: MIT Press.

Rescorla, M. 2015. The Computational Theory of Mind. *Stanford Encyclopedia of Philosophy.*

Rey, G. 1986. What's Really Going on in Searle's Chinese Room. *Philosophical Studies*, 50, 169-185.

Rey, G. 2002. Searle's Misunderstandings of Strong AI. In J. Preston and M. Bishop (Eds.), *Views* into the Chinese Room, 201-225. Oxford: Oxford University Press.

Schneider, S. 2011. *Language of thought: A new philosophical direction.* Cambridge, MA: MIT Press.

Scheutz, M. 2001. Causal versus Computational Complexity. *Minds and Machines,* 11, 534-566.

Searle, J. 1990. Is the Brain a Digital Computer? *Proceedings and Addresses of the American Philosophical Association*, 64, 21-37.

Searle, J. 1999. The Chinese Room. In R.A. Wilson and F. Keil (Eds.), *The MIT Encyclopedia of the Cognitive Sciences.* Cambridge, MA: MIT Press.

Searle, J. 2002. The Problem of Consciousness. *Consciousness and Cognition*, 2, 310-319.

Smart, J.J.C. 1959. Sensations and Brain Processes. *Philosophical Review*, 68, 141-156.

Turing, A.M. 1936. On Computable Numbers, with an Application to the Entscheidungsproblem. *Proceeding of the London Mathematical Society*, 42, 230–265.