Minds, Machines, and Turing

Steven Dao

Special Honors in the Department of Philosophy

The University of Texas at Austin

May 2015

Miriam Schoenfield

Thesis Supervisor

Sinan Dogramaci

Second Reader

Minds, Machines, and Turing

Steven Dao

May 2015

Since Alan Turing's introduction of a computation model now known as the *Turing machine*, philosophers of mind have asked if a Turing machine could provide an accurate and complete model of *mental* activity (Hodges). The *mechanists* believe that the mind's operation is equivalent to the operation of some Turing machine. In the 1960s, J. R. Lucas sought to show that this was an impossibility by appealing to Gödel's incompleteness theorems, two celebrated results that limit the theorems that can be derived in a consistent formal system of arithmetic. Later in that decade, Paul Benacerraf would build on Lucas's work, showing that mechanism was *not* an impossibility, though formally proving that one's own mind was equivalent to a particular machine was still impossible.

Lucas and Benacerraf restricted themselves to discussing simulations of the mind that were based on *formal systems of classical first-order logic*[1]. In this paper, I will argue that formal systems are poor substitutes for Turing machines, and so Lucas's and Benacerraf's results actually tell us very little about what can or cannot be done by Turing machines. I will then attempt to carry out Lucas's and Benacerraf's anti-mechanist program by directly examining Turing machines that purport to simulate minds. My eventual conclusion is that minds cannot be simulated by Turing machines. I then explore the consequences of this claim.

[1]Throughout this paper, the terms "first-order logic" and "formal system" will generally refer to classical theories of first-order logic.

1 The Mechanists and the Anti-Mechanists

Before I dive in, I will give a brief overview of the anti-mechanist arguments given by Lucas and Benacerraf in their original papers. Although they are somewhat problematic, Lucas's and Benacerraf's arguments provide a good jumping-off point for our anti-mechanist program.

In his 1961 paper *Minds, Machines, and Gödel*, J. R. Lucas attempts to refute mechanism by appealing to Gödel's incompleteness theorems. From here on out, I'll assume that *mechanism* refers to this general thesis:

> **M**. For any mind, there exists some Turing machine that simulates that mind's cognitive system.

I'll touch on what exactly *simulating the mind's cognitive system* involves in a moment. For now, let's just stick to the general thesis above. Lucas looks to refute this mechanist position, that is, he seeks to show that this general anti-mechanist thesis holds:

> **A**. There exists some mind for which no Turing machine simulates that mind's cognitive system.

Now that the preliminaries are out of the way, let's turn to the pressing issue—what exactly does "simulating the mind's cognitive system" entail? Lucas himself is not very explicit in his paper. He states that "any mechanical model of the mind must include a mechanism which can enunciate truths of arithmetic, because this is something which minds can do" (Lucas 115). Unfortunately, his terminology leaves unclear how minds exactly "enunciate" the truths of arithmetic. He suggests that "enunciating" means being able to "see" that a sentence is true (120), being able to "produce" it as true (115), or being able to "convince" oneself of a sentence's truth (115). However, this is rather vague at best—couldn't I convince myself of anything? Benacerraf points out that Lucas does not

mean "enunciating" or "seeing" as *"proving in a formal system"*: "[Lucas] seems content to allow the sense in which he can prove things that [a formal system] cannot to remain an informal one" (Benacerraf 20). Benacerraf interprets Lucas as thinking that simulating the mind's cognitive system requires outputting its body of knowledge[2]. By this interpretation, we should assume that **M** implies the following:

> $\mathbf{M_K}$. For any mind, there exists some Turing machine that outputs all and only the sentences known by that mind.

In turn, this means that the anti-mechanist thesis **A** is a consequence of the following:

> $\mathbf{A_K}$. There exists some mind for which no Turing machine outputs all and only the sentences known by that mind.

Lucas and Benacerraf don't give much reason for thinking that this interpretation of "simulating a cognitive system" is the right one. Nevertheless, we'll assume that this interpretation works, at least for the time being, so that we can discuss the rest of the argument in more detail. So for now let's just take for granted that **M** implies $\mathbf{M_K}$.

Lucas's argument focuses on the arithmetical capabilities of the mind instead of examining everything that the mind knows. That is, he focuses on proving this claim:

> $\mathbf{A'_K}$. There exists some mind for which no Turing machine can output all and only the sentences of arithmetic known by that mind.

Now he focuses on $\mathbf{A'_K}$ specifically because it would imply $\mathbf{A_K}$ as well. This seems

[2]The argument presented in (Benacerraf 22–29) also assumes knowledge, rather than belief, because it deals with minds "proving" sentences of arithmetic. However, a different (perhaps more charitable) reading of Lucas suggests that he thinks simulation requires outputting a body of *beliefs* rather than a body of *knowledge*, since he has concerns about the mind's consistency (Lucas 121), something that does not affect a body of knowledge. Indeed, there are problems with requiring knowledge as opposed to beliefs, as Section 2 of this paper will show. Regardless, there are other shortcomings in Lucas's and Benacerraf's arguments that will hinder them either way.

plausible: suppose that $\neg\mathbf{A_K}$: for any given mind, there *is* a Turing machine simulating that mind's body of knowledge. (That is, suppose that the mechanist thesis $\mathbf{M_K}$ holds.) But then we could construct another machine that simply filters out only the arithmetical things that the mind knows. The new machine could simply go through all of the things output by the original machine, re-outputting only those sentences that discuss arithmetic[3]. And thus we would have a Turing machine that can output all the arithmetical things that the mind knows, that is, we would have $\neg\mathbf{A'_K}$. Having shown the contrapositive, we can see that, were $\mathbf{A'_K}$ to hold, $\mathbf{A}$ must hold as well.

Now even though Lucas claims that he's arguing against *Turing machine* simulations of the mind he ultimately discusses *formal systems*, not Turing machines. That is, his paper ends up giving an argument not for $\mathbf{A'_K}$ but for a more specific claim:

> $\mathbf{A''_K}$. There exists some mind for which no theory of classical first-order logic can derive all and only the sentences of arithmetic that that mind knows.

Lucas assumes, without any real explanation, that $\mathbf{A''_K}$ will imply $\mathbf{A'_K}$ (which, in turn, implies $\mathbf{A_K}$, the root mechanist thesis that he's trying to prove). Again, I'll give Lucas the benefit of the doubt and continue with the assumption that $\mathbf{A''_K}$ implies $\mathbf{A'_K}$ in mind. However, I'll later show that this assumption is another sticking point in his argument.

Let's continue with Lucas's argument. Now the mind knows quite a bit of arithmetic—from addition and subtraction to multiplication and division. This, along with a few basic rules of arithmetic that most people know, is enough to define Robinson arithmetic[4], a system of first-order logic in which Gödel's incompleteness theorems apply[5].

[3] If you're not convinced by this, wait until Section 3, which discusses Turing machines in greater depth.

[4] I've specified Robinson arithmetic here as opposed to Peano arithmetic because PA involves induction, a mathematical principle generally unknown to the layman. In contrast, Robinson arithmetic doesn't have induction, even though it is susceptible to Gödel's theorems. See *Computability and Logic* (Boolos, Burgess, and Jeffrey), for a discussion of Robinson arithmetic. Note that, although most literature uses **Q** to denote Robinson arithmetic, the Boolos text uses **Q** to denote a similar system and uses **R** for Robinson (216).

[5] Note that Robinson arithmetic is already incomplete without resorting to Gödel's incompleteness theorems because it cannot prove theorems that require mathematical induction.

Gödel's incompleteness theorems are two theorems of logic that apply to formal systems possessing a certain amount of arithmetical power, such as Peano or Robinson arithmetic. The first theorem shows that, if such a system were consistent, then there would exist some sentence G for which the system would derive neither G nor $\neg G$. And yet, were such a system consistent, then the sentence G would be true, though not provable in the same system. We'll call the sentence G the system's *Gödel sentence*.

According to Lucas, because a formal system modelling the mind must derive all the truths of arithmetic that a mind knows, any such system must be able to derive the truths of Robinson arithmetic at the least, meaning that it is susceptible to Gödel's incompleteness theorems. So the system could not prove its Gödel sentence G.

Yet Lucas claims that a mind could know the sentence G to be true were it to know that the system was consistent, meaning that the mind would know something that the system can't, and thus the formal system could not be a complete simulation of the mind. Thus it seems that, having proven $\mathbf{A}''_{\mathbf{K}}$, we prove $\mathbf{A}'_{\mathbf{K}}$ and $\mathbf{A}_{\mathbf{K}}$ and $\mathbf{A}$. However, the argument is a lot less powerful than it initially seems. Lucas only shows that *if the mind knows that the system is consistent*, then the mind can know the sentence G to be true. So everything hinges on whether the mind can know the consistency of the relevant formal system. The argument does seem a bit question-begging: if the mind can do something that a formal system can't then the mind is more powerful than the formal system simply by definition. But we must give Lucas credit here: he has identified that the anti-mechanist can "beat" the mechanist by (informally) coming to know that the system is consistent, something that *intuitively* seems to be in the realm of possibility.

But to this last piece of the puzzle, Lucas does not give a very satisfying reason *why* we should believe that we can know the formal system's consistency, other than this appeal to intuition. Paul Benacerraf points out that very intelligent logicians have come up with formal systems whose axioms were thought to be consistent, only to be later

proven wrong[6]. It may turn out that well-known systems such as Robinson or Peano arithmetic are inconsistent and we simply haven't found proofs of their inconsistency yet.

It is not at all clear that I can know the consistency of *any arbitrary formal system*. In *God, the Devil, and Gödel* (1967), Benacerraf proposes an alternative version of Lucas's argument that begins with weaker premises, ending with a weaker conclusion that is still interesting in and of itself.

Benacerraf's proof is essentially a *reductio* argument that shows the conjunction of the following statements to be false[7]:

There exists a Turing machine W such that

1. I know that W outputs enough arithmetic in which to carry out Gödel's theorems, e.g. it derives the theorems of Robinson or Peano arithmetic,

2. I know that the sentences that W outputs are a subset of the arithmetical sentences that I know to be true, and

3. the sentences that W outputs are a superset of the arithmetical sentences that I know to be true, i.e., W proves everything that I know.

So for any given Turing machine W, at least one of these sentences must be false by Benacerraf's proof. If we reject (3) then we show something similar to Lucas's original argument: there is something I know that ends up not being a theorem of W. If, on the other hand, I reject (1) or (2), I leave open the possibility that I *am* simulated by W, but I simply cannot know so. Simply put, either my mind cannot be modelled by a Turing machine or "perhaps I cannot ascertain which one" (Benacerraf 29).

[6]For example, the famed logician and philosopher Alonzo Church developed an early system of "illative" λ-calculus that turned out to be inconsistent (Alama). If a logician as knowledgeable as Church could be wrong on the consistency of a formal system, who are we to claim that we could fare better?

[7]See step 9 of his argument in *God, the Devil, and Gödel* (Benacerraf 25).

But Benacerraf's adjustments are not enough to save Lucas's argument. Both Benacerraf and Lucas are hinging everything on two debatable assumptions that I previously pointed out: (1) the assumption that $\mathbf{A_K}$ implies $\mathbf{A}$ and (2) the assumption that $\mathbf{A''_K}$ implies $\mathbf{A'_K}$.

Let's consider (1) the assumption that $\mathbf{A_K}$ (there exists some mind whose knowledge corresponds to no Turing machine's output) implies $\mathbf{A}$ (there exists some mind whose cognitive system can't be simulated by any Turing machine). This is the same as saying that $\mathbf{M}$ implies $\mathbf{M_K}$. It's not at all clear why this is the case, and we have no affirmative reason for believing so. Sure, knowledge is somewhat related to cognition, but it also requires external conditions to obtain in a way that belief does not. Why is it that, were we able to simulate any mind's cognitive system via a Turing machine, we would also be able to output all and only that mind's knowledge via a Turing machine?

The latter requires quite a bit of extra information compared to the former. If a Turing machine simulated my mind's knowledge, it would be able to predict many things about the future and the state of the world. I believe "The sun will rise tomorrow", so if the machine outputs that statement, it will perfectly predict that the sun will rise tomorrow. I believe "Peano arithmetic is consistent", so if the machine outputs that statement, it will perfectly verify that Peano arithmetic is indeed consistent. A machine capable of such deductions seems super-powerful, perhaps too powerful, and the mechanist would be wary of claiming that it exists.

Being able to output a mind's knowledge would also require that the Turing machine knows things that not even the mind itself would know. Since this machine must be able to discern true beliefs from false ones, unjustified beliefs from justified ones, and Gettier cases from non-Gettier cases, it will end up knowing what you do not know. But it's uncontroversial that, even when introspecting, you don't know what you do not know. The mechanist wouldn't commit to saying that simulating a mind requires a machine

stronger than the mind itself!

If we assume that $\mathbf{M}$ implies $\mathbf{M_K}$, then we would be committed to saying that, were we able to simulate all minds' cognitive systems via Turing machines, we would also give rise to a class of bizarre super-powerful Turing machines. I think the mechanist would be hesitant to accept any argument that requires this assumption. It would mean that the simulating machines he must produce are somehow ultra-powerful. I'll leave off by saying that it's not logically impossible that $\mathbf{M}$ implies $\mathbf{M_K}$. However, it just seems a bit strange to think that's the case.

Now let's move on to (2) the assumption that $\mathbf{A''_K}$ (there exists some mind whose arithmetical knowledge corresponds to no theory of classical first-order logic) implies $\mathbf{A'_K}$ (there exists some mind whose arithmetical knowledge corresponds to no Turing machine's output). Remember that Benacerraf's argument (Benacerraf 23–29) is structured wholly in terms of *formal systems* and not in terms of actual *Turing machines*. Lucas does the same thing throughout his paper. They have both equated the output of a Turing machine to the theorems that can be derived in a formal system of classical first-order logic.

This is a sticking point in both of their arguments. In an argument about Turing machines, it will not help to use formal systems as proxies. Paul Benacerraf argues that discussing formal systems in place of Turing machines "is legitimate, since for any system which is formal in the required sense[8], there exists a theorem-proving machine which 'proves' all and only the theorems of that system" (Benacerraf 13). I agree that this is true, but for Lucas's and Benacerraf's argument to apply to *all Turing machine simulations*, it seems that they actually need the converse: for any Turing machine, there exists a formal system whose theorems correspond to what the Turing machine proves.

But they do not have the converse. Consider a Turing machine that prints out all well-formed sentences of classical first-order logic that are less than ten characters long. This

[8]The required sense being a finitely-axiomatizable theory of classical first-order logic.

Turing machine will print out a finite set of sentences that includes the sentence $\exists x(x \neq x)$. There is no corresponding formal system that can contain exactly these sentences, since a contradiction in classical first-order logic will imply all other sentences[9].

Let's return to Lucas's original argument. If we accept it despite its shortcomings, we will obtain $\mathbf{A}''_{\mathbf{K}}$ at the very least. But this is a claim about formal systems, and not about Turing machines. And since there are Turing machines such as the less-than-ten-symbols machine whose output does not correspond to a formal system of classical first-order logic, $\mathbf{A}''_{\mathbf{K}}$ most certainly *does not* imply $\mathbf{A}'_{\mathbf{K}}$.

And so the chain is broken. Lucas's argument, even if we are to accept the core of it, cannot give us $\mathbf{A}$ at the end, no matter which way we spin it. The mechanist has won this battle. But can we win the war using a different strategy? Let's use the lessons learned from Lucas and Benacerraf to construct a new argument that will fare better. We'll remember the two mistakes that Lucas and Benacerraf made: (1) assuming that a simulation of the mind's cognition necessitates a simulation of the mind's knowledge and (2) assuming that formal systems can be proxies for Turing machines.

2 A Different Strategy

In Section 1, I discussed Lucas's and Benacerraf's arguments, which revolve around a simulation of the mind's arithmetic knowledge. As we've seen, their arguments for $\mathbf{A}$ have several shortcomings that are not easily rectified.

As I've shown before, Lucas's and Benacerraf's strategy of trying to prove the claims $\mathbf{A}''_{\mathbf{K}}$, $\mathbf{A}'_{\mathbf{K}}$, and $\mathbf{A}_{\mathbf{K}}$ is not the most solid avenue for proving $\mathbf{A}$[10]. But their strategy is sound

[9] A paraconsistent theory would be able to deal with sentences such as $\exists x(x \neq x)$ without exploding, as opposed to a classical theory. However, it's clear that Lucas and Benacerraf are referring to classical theories of first-order logic. Paraconsistent theories might be another way to go in correcting Lucas's and Benacerraf's original arguments, but they are outside of the scope of this paper.

[10] Again, although it's not logically impossible that $\mathbf{A}_{\mathbf{K}}$ implies $\mathbf{A}$, our opponent the mechanist might be wary of accepting this assumption. For the rest of my argument, I try to limit myself to premises I think the mechanist would willingly accept.

in one respect—rather than trying to disprove **M** (i.e. prove **A**) directly, which would be a daunting task indeed, they break it up into a series of simpler inferences. So here we want to find a different path that will simplify our job of disproving **M** while keeping us on solid ground.

Let's first review the mechanist thesis **M**:

> **M**. For any mind, there exists some Turing machine that simulates that mind's cognitive system.

And the corresponding anti-mechanist thesis **A**:

> **A**. There exists some mind for which no Turing machine simulates that mind's cognitive system.

We'll need to pinpoint what we mean by *simulating the mind's cognitive system* if we are wary of saying that it means "outputting the mind's body of knowledge". Something that's clearly related to the mind's cognitive system is its body of beliefs. Beliefs are intimately tied to cognition, but they don't depend on external conditions obtaining, as knowledge would. I don't have any qualms in claiming that the simulating the mind's cognitive system requires, at the very least, outputting its body of beliefs[11]. By this interpretation of "simulating the mind's cognitive system", we should assume that the broad thesis **M** implies the following instead:

> $\mathbf{M_B}$. For any mind, there exists some Turing machine that outputs all and only the beliefs of that mind.

In turn, this means that the broad anti-mechanist thesis **A** should now be understood as

[11] One reading of Lucas suggests that this is what he actually meant. Lucas devotes much space to consistency in his paper, suggesting that he's actually discussing beliefs, or something close to beliefs, rather than knowledge. See Footnote 4 for more information.

a consequence of the following:

> **$\mathbf{A_B}$.** There exists some mind for which no Turing machine can output all and only the beliefs of that mind.

It definitely seems more plausible that **M** implies $\mathbf{M_B}$ and that $\mathbf{A_B}$ implies **A**. We can accept that **M** implies $\mathbf{M_B}$ without the claim **M** now committing us to the existence of super-powerful Turing machines, so the mechanist should have no reason to object. (Compare this to the old case where we interpreted "simulating the mind's cognitive system" as requiring the output of knowledge instead of belief; in that case, we would be forced to accept super-powerful Turing machines in the case that **M** were true.) Thus it's the anti-mechanist claim $\mathbf{A_B}$ that I'll focus on proving in the rest of this paper. If I can successfully prove $\mathbf{A_B}$, I will have proven **A** as well.

Here, when I discuss beliefs, I take it that minds have *infinitely many beliefs*. I consider the set of all beliefs, not just the beliefs that one has consciously and explicitly contemplated. For example, a mind might believe that 1,593,932 is a number because it knows about the decimal number system, even though it has never consciously contemplated this fact before.

We will now attack $\mathbf{A_B}$ directly with Turing machines rather than try to prove a claim about formal systems, such as

> **$\mathbf{A'_B}$.** There exists some mind for which no theory of classical first-order logic can derive all and only the beliefs of said mind.

Such a strategy would be unwise, as we have seen from Lucas's and Benacerraf's example. I digress a bit and claim that $\mathbf{A'_B}$ is trivially true, and the mechanist would certainly agree that it is true. The set of things I believe is quite different from the set of theorems in a formal system of arithmetic: the former are not closed under consequence while the

latter surely must be. Thus the set of things I believe can be partially inconsistent, while a system of arithmetic is either completely consistent or completely inconsistent. Certainly I'm not a total idiot who would believe outright contradictions (such as "Ringo has exactly five dogs" and "Ringo has exactly four dogs"). But my beliefs contain contradictions formed by several statements that, pairwise, are not outright contradictions but when combined are inconsistent. For example, a reasonable person might have the following beliefs:

1. "Paul says only true things."
2. "George says only true things."
3. "Paul says that Ringo has exactly five dogs."
4. "George says that Ringo has exactly four dogs."

If we represented these four statements in a formal system of classical first-order logic, we would inevitably produce *all well-formed sentences* of the system due to the rules of classical first-order logic. But this doesn't adequately explain how the mind's beliefs work. Sure, I might believe the four mutually-inconsistent statements above. But that doesn't mean that I would believe "2+2=5" as a consequence. That's just absurd.

I won't explode like a formal system of classical first-order logic and somehow just believe every possible sentence (Lucas 121)! I agree with Lucas when he claims that although we may have some inconsistencies, "when a person is prepared to say anything, and is prepared to contradict himself without any qualm or repugnance, then he is adjudged to have 'lost his mind'". We should expect beliefs to be somewhat inconsistent, although never completely inconsistent in sane minds.

As I'll demonstrate in Section 3, *Turing machines* can represent selective inconsistency without exploding. A Turing machine could certainly encode the four inconsistent beliefs above and then stop without encoding anything else to its tape.

So a mind with these beliefs would prove to us $\mathbf{A'_B}$ (there exists a mind for which no formal system can derive exactly the beliefs of said mind) since no theory of classical first-order logic can derive exactly the above four statements. And yet it would not prove that $\mathbf{A_B}$ (there exists a mind for which no Turing machine can output exactly the beliefs of said mind) since there indeed could be a Turing machine that outputs exactly the above four statements.

The non-equivalence of Turing machines and classical first-order logic shows us that, in the same way that Lucas's $\mathbf{A''_K}$ did not imply his $\mathbf{A'_K}$, our claim $\mathbf{A'_B}$ will not help us prove $\mathbf{A_B}$. In Section 4, I'll give a proof for $\mathbf{A_B}$ directly instead. And recall that, once we have a proof for $\mathbf{A_B}$, we will have proven the broad anti-mechanist thesis $\mathbf{A}$.

3 Turing Machines

Up to now, I've been discussing Turing machines in an informal way. I've treated them mainly as black boxes that somehow output sentences. But for the rest of this paper, it will be extremely helpful for me to define Turing machines in more detail. In this interlude, I'll give a brief overview of Turing machines for those who are unfamiliar with them. (Those with a basic background in Turing machines can feel free to skim over the next few paragraphs.)

There are several compatible definitions of a Turing machine, but I'll adopt this definition given by Barker-Plummer. A Turing machine has a one-dimensional tape, partitioned off into separate cells. The tape is finite on one end (say, the left end) and infinite on the other end (say, the right end). Each cell contains either the symbol `1` or the symbol `0`. To represent a sentence on the tape, it must be encoded in a binary format using only the symbols `0` or `1`. (So if I say that something is printed or written to the tape, I specifically mean that it is *encoded in binary* and then placed on the tape.) The Turing machine also has a "head" that rests on a single cell of the tape at a time. Finally, the Turing machine

has a finite list of states controlling its behavior, which we may number state 1, state 2, ..., state n. At any given time, the Turing machine must be in one of these states.

I'll refer to the combination of the tape contents, the head position, and the state as the Turing machine's *configuration*. From a starting configuration, the Turing machine's behavior is deterministic, acting according to a list of *transition rules*. Each rule specifies the actions that the Turing machine will take when it is in a certain configuration. The actions the Turing machine can take include moving the head, writing a symbol to the tape position underneath the head, or changing the current state. Some configurations may be *halting* or *stopping* configurations, from which the Turing machine has no defined transition and so stops acting. A Turing machine that never enters a halting configuration is said to *never halt* or to *loop forever*.

Although the work of the Turing machine seems basic at first, Turing machines can perform complex tasks such as multiplication through composition of these basic actions. Basically anything that can be done on a digital computer (ignoring external input) can be performed on a Turing machine. A Turing machine can "branch" between two different pathways in its program depending on the contents of its tape. In addition, a Turing machine can also simulate the behavior of another Turing machine[12]. Both of these characteristics play an important role in the rest of this paper.

It's also important to note that there are a *countably infinite* number of Turing machines. A consequence is that each Turing machine can be assigned a natural number as its *code number*; for a machine Π, I'll denote its code number as $\langle\Pi\rangle$. (The system for deriving code numbers from Turing machine definitions is not too important, but imagine that we do something similar to a Gödel-numbering system used in proofs of Gödel's incompleteness theorems.)

From here on out, I'll express the *configuration* and the *transition rules* of Turing machines in plain English so that the argument is easier to follow. Rest assured that I will

[12]See the discussion on universal Turing machines in Barker-Plummer's article "Turing Machines".

limit my plain-English descriptions to actions that we could actually encode in a formal description of a Turing machine.

Before I delve into the intricacies of using Turing machines to simulate the mind, I must first define some helpful terminology. When I say that a Turing machine *enumerates* a set of sentences (for a given input), I mean that the machine encodes each sentence onto its infinite tape, one by one, going from left to right. Each sentence in the set will be encoded to the tape after some finite amount of time, though the machine will run forever if the set is infinite. When I say that a Turing machine *accepts* a sentence, I mean that, when the machine is started with the sentence encoded on its tape, it eventually stops and encodes "yes" onto the tape. When I say that a Turing machine *rejects* a sentence, I mean that, when the machine is started with the sentence encoded on its tape, it eventually stops and encodes "no" onto the tape. And when I say that a Turing machine *accepts* a set of sentences, I mean that the Turing machine accepts all and only those sentences in the set. (The Turing machine need not stop for a sentence outside of the set.)

Recall from Section 1 that there exist some sets of sentences that we can represent using the output of a Turing machine but not as theories of classical first-order logic. I previously gave the example of a set containing all sentences of classical first-order logic that are less than ten symbols long. This set was finite, and you might point to this fact to explain why some Turing machines have outputs that don't correspond to any formal system's theorems. After all, any formal system of classical first-order logic will have infinitely many theorems (it must include, at a bare minimum, the logical tautologies).

But this can happen with infinite sets as well. Consider the set composed of the sentence "$\exists(x \neq x)$" along with all the theorems of Presburger arithmetic. (Presburger arithmetic is a weak system of arithmetic that can prove its own consistency, unlike Robinson or Peano arithmetic. It is too weak to carry out Gödel's incompleteness theorems (Weisstein).) Presburger arithmetic by itself is a theory of classical first-order logic, and as a

theory of classical first-order logic, it must be semi-decidable, so there must exist some Turing machine that *enumerates* all and only the theorems of Presburger arithmetic. Call this enumerator-machine C. Then we can provide another enumerator-machine for the set of all theorems of Presburger arithmetic *plus* the statement "$\exists x(x \neq x)$". Let's call this second machine C':

1. C' first encodes the statement "$\exists x(x \neq x)$" to the tape and moves the head past the end of the encoded statement.

2. C' then runs the machine C and outputs to its own tape whatever C outputs.

So while formal systems of classical first-order logic can't represent this set, a Turing machine certainly can certainly enumerate the set!

So again it's clear why Lucas's and Benacerraf's arguments didn't work out. Negative claims about formal systems, such as $\mathbf{A''_K}$ (there exists a mind for which no formal system derives the arithmetic said mind knows), will not necessarily translate into negative claims about Turing machines, such as $\mathbf{A'_K}$ (there exists a mind for which no Turing machine outputs the arithmetic said mind knows).

Now that we have a clear idea of what Turing machines are capable of doing, we are in a better position to understand the mechanist thesis $\mathbf{M_B}$. Recall that $\mathbf{M_B}$ is the claim that, for any mind, there exists some Turing machine that outputs all and only the beliefs of that mind. By "outputting" here, I mean that the Turing machine *enumerates or accepts* exactly the sentences that the mind believes. When the Turing machine enumerates the mind's beliefs, I'll assume that all of these can be encoded in some manner that can be represented with the symbols `0` and `1` on the Turing machine's tape. (For example, a mind whose inner voice happens in English might have its beliefs encoded using some kind of Gödel numbering system for the Latin alphabet.)

3.1 Enumerator-Machines

It's thus quite natural to view the simulation as an enumerator-machine that simply spits out the beliefs of the mind, one encoded snippet at a time. That is, we will naturally expect an enumerator-machine that simulates a person, Patty, to do the following:

1. When started, the machine will ignore whatever is on its tape.
2. The machine proceeds to replace the contents of its tape with an encoding of Patty's beliefs, one by one, sentence by sentence.
3. The machine will never halt, since Patty has infinitely many beliefs.

Now, however intuitive the enumerator-machine model may be, it might not be the best one to consider in our arguments. At times it will be easier to structure our arguments in terms of the equivalent *acceptor-machines*. For this reason, I'll give a brief picture on how we can define acceptor-machines and how they're related to the intuitive enumerator-machine model.

3.2 Acceptor-Machines

A machine that simulates the mind's beliefs could do so by *accepting* things that the mind believes, while *doing nothing* for things that do not form part of the mind's beliefs. We will expect such a machine, simulating a person, Patty, to do the following:

1. Let S be a statement encoded on M's tape when it starts up.
2. When started with a statement S encoded on its tape such that Patty believes S, the machine should stop and write "yes" to the tape.
3. When started with a statement S encoded on its tape such that Patty does not believe S (e.g. Patty disbelieves or suspends judgment on S), the machine should run forever without halting.

In the acceptor-machine model, you cannot actually determine whether a certain statement S is one of Patty's beliefs. Let's take an example where we'd like to determine whether Patty possesses the belief S. If she does indeed believe it, the acceptor-machine will answer "yes". But if she disbelieves S or suspends judgment on S, then the acceptor-machine will run forever. At no point can you know whether the machine will eventually stop, and thus you couldn't determine in a known, finite amount of time whether Patty actually believes S.

That this model is quite weak in this regard is one reason that the mechanist should be eager and willing to accept it as what it means to simulate a mind via Turing machine. Ultimately, I think this is the minimum work that the mechanist must do in order to provide a machine that would be adequate for demonstrating the thesis **M**. I don't think we're making any demands here that the mechanist would find objectionable. In fact, we as the anti-mechanists might be even putting ourselves at a slight disadvantage by committing to this relatively-weak model of simulation.

But one useful aspect of the weak acceptor-machine is that it is at least equivalent in power to the enumerator-machines that we have already seen. If we have an acceptor-machine A, we can easily produce an enumerator-machine:

1. The enumerator-machine keeps a counter, i, that is initially set to 1.

2. The enumerator-machine iterates through the first i strings in lexicographic order. For example, if using binary strings, it would enumerate through the list 0, 1, 01, 10, 11, 000, 001, ... for the first i strings.

3. For each of the i strings, the enumerator-machine runs the machine A for i steps with the string encoded on its tape. If the machine A accepts the string, then the enumerator-machine will print the string to its own tape.

4. The enumerator-machine increments i by one.

5. The enumerator-machine repeats steps 2–4.

Similarly, if we have an enumerator-machine B, we can produce the corresponding acceptor-machine:

1. The acceptor-machine is started with some string S encoded on its tape.
2. The acceptor-machine runs B until it sees that B has printed S onto its tape.
3. If the acceptor-machine sees that B has printed S, then the acceptor-machine will stop and print "yes" to its own tape. Otherwise, the acceptor machine will have to continue running forever.

For our purposes, we can now interchange "acceptor-machines" and "enumerator-machines": whenever one kind is required, the other can act as a perfect substitute.

Now that we've gone over the appropriate background in Turing machines, let's get back to the argument at hand. In the next section, I'll execute my strategy of directly proving $\mathbf{A_B}$, that there exists a mind whose beliefs correspond to no Turing machine.

4 Yet Another Anti-Mechanist Argument

In Section 2, I claimed that a comprehensive simulation of the human mind, of the kind postulated by the mechanist in his claim **M**, must represent, at the very least, a person's beliefs. So if we were to show that $\mathbf{A_B}$, that is, there exists a mind whose beliefs cannot be exactly output by a Turing machine, then we will have defeated the mechanist.

In this section, I'll present the case for $\mathbf{A_B}$ that I've been building up to thus far in this paper. I believe that, once the argument is done, it will be difficult for the mechanist to continuing advocating his position. Let me restate $\mathbf{A_B}$:

$\mathbf{A_B}$. There exists some mind for which no Turing machine can output all and only the beliefs of that mind.

To start off, let's consider a simple *counting argument* for $\mathbf{A_B}$. Consider that the set of all possible beliefs β is at least countably infinite. For example, "x is a number" for any integer x is a possible belief, meaning that there are a countably infinite number of possible beliefs with this template.

Now we see that a specific mind's body of beliefs β' is a subset of the possible beliefs β. All of the beliefs that I possess will come from the set of all possible beliefs, although I (and most others) will only believe some of the possible beliefs. Consider that the set of possible bodies of belief $\{\beta', \beta'', \beta''', ...\}$ is the set containing all subsets of the possible beliefs β.

But according to Cantor's theorem, the set containing all subsets of an infinite set must, in turn, be uncountably infinite[13]. So the set of possible bodies of belief is uncountably infinite as well. However, there are only countably many Turing machines. (Consider that each Turing machine must have a finite definition, and there are only countably many strings of arbitrary, finite length.) Thus there are more possible bodies of belief then there are Turing machines. If we attempted to assign a Turing machine to each possible body of belief, we would eventually run out of Turing machines before exhausting all possible bodies of belief. Thus, $\mathbf{A_B}$ holds: there must exist minds whose bodies of belief do not correspond to any Turing machine's output.

Now the question is, *what are these minds even like*? The counting argument is a bit unsatisfying without knowing exactly which minds possess beliefs that correspond to no Turing machines at all. So to make the argument more clear, I'm going to give an example of a mind whose beliefs cannot be simulated by a Turing machine.

Somehow, I find myself back where I started: at Lucas's original argument that mechanism is false. In order to show what a non-simulatable mind looks like, I'll derive a spiritual counterpart to Lucas's original anti-mechanist argument, but of course I'll argue

[13] A mathematical example: the integers are countably infinite, so the set of all subsets of the integers is, in turn, uncountably infinite.

against Turing-machine mechanism directly as opposed to using formal-system mechanism as a stand-in. Whereas the demanding premises of Lucas's argument were its downfall, my argument's premises will prove shockingly easy to accept.

Recall that Lucas claims that the mind is more powerful than any formal system containing at least Peano or Robinson arithmetic because, for any such system, the mind can know whether the system is consistent, meaning that the mind could know the system's Gödel sentence. Thus the mind would be able to know a truth of arithmetic—the system's Gödel sentence—that the system would itself be incapable of proving. However, Lucas's assumption that we can know whether the system is consistent remains controversial. Lucas assumes in the first place that the mind is more powerful than the formal system, claiming that the mind can know the system's consistency. This premise is quite strong and not at all obvious. Thus it will be less controversial if we treat Lucas's argument as a conditional: *if Patty knows the consistency of a formal system*, then Patty's arithmetic power cannot be simulated by the formal system.

I'll also be claiming a conditional, only about Turing machines instead. Let M be some Turing machine that purports to accept all and only Patty's beliefs. (To simplify this argument, I'll consider M using the acceptor-machine model rather than the equivalent enumerator-machine model. But of course there is no practical difference to using one model over the other.)

The argument will involve a machine M', whose definition is based off of M, the machine that purports to simulate Patty's beliefs. Let M' be a Turing machine derived from M in the following manner:

1. When it starts, M' interprets the contents of its tape as the code number $\langle \Pi \rangle$ for some (arbitrary) Turing machine Π. (If the input is not a properly-encoded number, the Turing machine will simply loop forever without stopping.)

2. M' runs M with the statement S_Π on M's tape, where S_Π is defined as "The Turing

machine with code number $\langle \Pi \rangle$ loops forever when started with the code number $\langle \Pi \rangle$ encoded on its tape"[14].

3. M' waits for M to finish. If M stops with "yes" encoded on its tape, then M' will stop with "yes" encoded on its tape as well. Otherwise, M runs forever, and so M' will run forever too.

If you are familiar with the halting problem, you'll see that this setup is quite similar to many proofs of the halting problem. We're interested in whether *M' will halt or loop forever when started with its own code number $\langle M' \rangle$ on its tape.*

Let's say very specifically that Patty is *correct* as to a statement S when she believes S if and only if S is true. That is, she either (1) believes S and S is true or (2) believes S is false or suspends judgment and S is false. I will demonstrate that if *Patty happens to be correct as to whether M' loops forever on its own code number*, then Patty's beliefs cannot be accepted by the machine M. Note that the antecedent in this case is much weaker than the antecedent required by Lucas's argument. Patty only needs to have a certain belief (or lack such a belief) and, by happenstance, be correct; she need not prove or deduce anything whatsoever. In other words, she does not need any *knowledge* of whether M' will halt.

The argument is as follows. Suppose M is some Turing machine such that Patty happens to be correct as to whether M' loops forever when started on its own code number. Note that Patty does not need to *know* any of this—she might be lacking justification for her beliefs, for example. Her beliefs (or lack thereof) must simply "match up" with the current state of affairs. Or she might even be suspending judgment, having no belief whatsoever as to S, and S happens to be false.

Let's now suppose for reductio that M does indeed simulate Patty's beliefs, i.e., M

[14]So if the input to M' were its own code number $\langle M' \rangle$, then $\langle \Pi \rangle$ would be instantiated by $\langle M' \rangle$, meaning that M' would query M about the statement $S_{M'}$, or "The Turing machine with the code number $\langle M' \rangle$ loops forever when started with the code number $\langle M' \rangle$ encoded on its tape".

accepts all and only Patty's beliefs. I'll show that this supposition is impossible given what we know about Patty.

Since Patty is correct about M' looping forever, there are two cases to consider. By my previous definition of "correctness", either (1) she believes M' loops forever and it does loop forever, or (2) she has no such belief and M' eventually halts.

1. Suppose that Patty believes that M' loops forever on its own code number $\langle M' \rangle$[15], and M' doesn't ever halt. That is, Patty believes the sentence $S_{M'}$ ("The Turing machine with code number $\langle M' \rangle$ loops forever when started with the code number $\langle M' \rangle$ encoded on its tape"), and that sentence is true.

   Since M simulates Patty's beliefs, it accepts the sentence $S_{M'}$ because Patty believes that sentence. That is, when M is started with the sentence $S_{M'}$ on its tape, M will stop and output "yes". So now consider what happens when M' is started with its own code number $\langle M' \rangle$ on its tape. It will first simulate the behavior of M with the statement $S_{M'}$ on the tape. By definition, since M stops and outputs "yes", M' will also stop and output "yes".

   But then we can see that the sentence $S_{M'}$ ("The Turing machine with code number $\langle M' \rangle$ loops forever when started with the code number $\langle M' \rangle$ encoded on its tape") is false because M' *does indeed* halt when started with the code number $\langle M' \rangle$ on its tape!

   So in this case we arrive at a contradiction. Thus we must conclude that M cannot simulate Patty's beliefs in the case that Patty believes the sentence $S_{M'}$ and her belief is correct.

2. Now suppose that Patty either disbelieves that M' loops forever on its own code number $\langle M' \rangle$, or she suspends judgment on the question, and M' indeed does even-

[15]Patty does not even need to know that M' is derived wholly from M. It is sufficient that she has this belief about a machine with the same definition as M'.

tually halt on its own code number. That is, Patty possesses no belief in the sentence $S_{M'}$ ("The Turing machine with code number $\langle M' \rangle$ loops forever when started with the code number $\langle M' \rangle$ encoded on its tape"), and that sentence is false.

Since M simulates Patty's beliefs, it does not accept the sentence $S_{M'}$ because Patty disbelieves or suspends judgment on $S_{M'}$. That is, when M is started with the sentence $S_{M'}$ on its tape, M will run forever without stopping. So now consider what happens when M' is started with its own code number $\langle M' \rangle$ on its tape. It will try to simulate the behavior of M with the statement $S_{M'}$ on its tape. But since M loops forever, $S_{M'}$ will continue simulating M forever, and thus $S_{M'}$ will never halt, either.

But then we can see that the sentence $S_{M'}$ ("The Turing machine with code number $\langle M' \rangle$ loops forever when started with the code number $\langle M' \rangle$ encoded on its tape") is true because M' *indeed loops forever* when started with the code number $\langle M' \rangle$ on its tape!

So in this case we also arrive at a contradiction. Thus we must conclude that M cannot simulate Patty's beliefs in the case that Patty has no belief in the sentence $S_{M'}$ and $S_{M'}$ turns out to be false.

When we combine the two cases, we see that if Patty is correct about M', that is, she believes $S_{M'}$ if and only if $S_{M'}$ is true, then the machine M fails to simulate her beliefs.

Again, the interesting point for this version of the argument is that the antecedent is quite weak: Patty only needs for her belief (or non-belief) of the statement $S_{M'}$ to match reality for M to be unable to simulate her beliefs. She does not require any knowledge or justification of any kind—she might even be correct by happenstance. We don't need to postulate that Patty is special or gifted in any way. Thus, if there were a person that believed all sentences of the form $S_{M'}$ correctly, for any machine M, even if they had no grounds for such beliefs, they could not be simulated by any Turing machine whatsoever.

Certainly, if we're talking about possible minds and not just actual minds, the antecedent of our conditional is easy to accept. So at the very least, in some possible world, there exists a mind for which no Turing machine can be found. And so $\mathbf{A_B}$ holds: there is some mind for which no Turing machine can output exactly that mind's beliefs. And thus **A** holds as well: there is some mind whose cognitive system is not simulated by any Turing machine! Alas we have struck down the mechanist thesis **M**.

Now the mechanist may wish to modify his position. Mechanism, he says, is not a claim about all possible minds but a claim specifically about actual minds. That is, the mechanist will want to shift the playing field towards this thesis instead:

> **N**. For any actual mind, there exists some Turing machine that simulates that mind's cognitive system.

I'm willing to play ball on this new turf. Although we have demonstrated the existence of a *possible* mind for which there is no simulating machine, we have not yet demonstrated the existence of such an *actual* mind. For all we know, there is something that prevents such minds from being actually instantiated.

But I believe the best explanation will swing in our favor and not the mechanist's. Let's consider three possibilities:

1. All actual minds are Turing-simulatable. Even though there is a possible mind that cannot be simulated, no such minds actually exist. This is the mechanist's revised claim **N**.

2. All actual minds are non-Turing-simulatable. Even though there are possible minds that can be simulated, no actual mind happens to be simulatable.

3. A disjunctive explanation: some actual minds are Turing-simulatable while others are not.

I think that, off the bat, we should reject scenario (1), where all *actual* minds can be simulated, even though there are *possible* minds that can't. For this scenario to play out, we will have to have some kind of explanation as to *why* all of the actual minds can be simulated by Turing machines.

We've already shown that there are possible minds that no Turing machine could simulate. But nobody has shown that *there must exist minds that some Turing machine does simulate.* We now have some concrete evidence that there must exist non-simulatable minds, but we don't have any concrete scientific proof that a mind has ever been simulated by a Turing machine. So even though there are a countably infinite number of Turing machines, it could happen that *none of them correspond to a mind, actual or possible,* in the first place! The mechanists speculate that minds are simulatable, but they haven't given us an example of one thus far.

Furthermore, we see now that there are an *uncountably infinite* number of minds that can't be simulated, while up to a *countably infinite* number of minds that can be simulated. Any explanation for scenario (1) would have to explain why our actual minds are so incredibly special that they can be simulated by Turing machines. The number of minds simulatable by Turing machines makes up an infinitesimally small proportion of all the minds that could possibly exist. In fact, suppose that actual minds are drawn from a uniform probability distribution over all the possible minds. The probability of picking a simulatable mind is zero, and the probability of picking a non-simulatable mind is one[16].

To gain a better appreciation of how remarkably bad this is for the mechanist, consider the following situation. The mechanist lines up a hundred people, and says that he has a hundred Turing machines, each of which will simulate a respective person's mind. For a person P_n (where $1 \leq n \leq 100$), let's say that their corresponding machine is M_n. For

[16]Probabilities involving uncountable sample spaces are tricky. For a uniform distribution on a sample space equinumerous with the real numbers (such as the sample space of the possible minds), the probability of a countable event is zero. An example: the probability that a uniformally-distributed random real number is an integer is zero because there are only countably many integers.

scenario (1) and the claim **N** to hold, that is, for the mechanist to tell the truth about all these Turing machines, all one hundred people from P_1 to P_{100} must have an incorrect belief as to the sentence $S_{M'_n}$ for their machine M_n. That's to say, *their beliefs must be worse than even random chance*, which states that, were all one hundred people to guess, half of them (or fifty people) would guess correctly. This isn't even remotely plausible. And yet it's the scenario that the mechanist is committed to were he to hold on to **N**.

Now, the mechanist might deny that the scenario above is bizarre. He might say that the very reason that all 100 people get the question wrong is precisely *because* they can be simulated by Turing machines. Determining the machine that simulates a person would depend on that person's beliefs, both correct and incorrect, in the first place. And these people already had the incorrect beliefs as to their specific $S_{M'_n}$, which plays into why the machine M_n simulates them. If they were correct, the mechanist's process for determining M_n would have simply picked a different machine. I admit that this rebuttal does serve to demonstrate that the above scenario might not be too strange at all.

However, another thing to consider is that, from the reductio proof given above, we can obtain a result similar in spirit to Paul Benacerraf's proof[17]. We can show that, for any Turing machine M, an introspective mind *couldn't justifiably believe* that it were simulatable by M. Suppose that Patty were an introspective person and that she believed that the machine M simulated her. She introspects to determine her beliefs regarding M and its corresponding machine M'. Her beliefs can be divided into three cases:

1. She believes that M simulates her and that M' loops forever (and knows that she believes this). She then deduces that M outputs the sentence $S_{M'}$ ("The Turing machine with code number $\langle M' \rangle$ loops forever when started with the code number $\langle M' \rangle$ encoded on its tape"). But then Patty deduces that M' *eventually halts*, leading to an inconsistent state of beliefs.

[17]Recall that Benacerraf proved that either no Turing machine simulates a mind or the mind cannot determine which Turing machine simulates it.

2. She believes that M simulates her and that M' eventually halts. From this, Patty deduces that M doesn't output $S_{M'}$. But then Patty deduces that M' *loops forever*, leading to an inconsistent state of beliefs.

3. She believes that M simulates her and suspends judgment on the question of M'. From this, Patty deduces that M' doesn't output $S_{M'}$, meaning that M' loops forever, as in case 2. However, further introspection leads to the deduction that M' eventually halts, as in case 1, leading to an inconsistent state of beliefs.

Whatever the case, Patty ends up able to deduce that M' both loops forever *and* eventually halts. It seems irrational to maintain this obviously contradictory state of beliefs. Patty could not justifiably believe something that leads to an obvious contradiction. So, even if M simulated her, Patty *couldn't have the justified belief* that it does. But it seems strange to think that one is unable to know something that were true. If scenario (1) held, the mechanist would need to be able to explain this nebulous barrier on your beliefs.

From all this evidence, it's hard to see why we should accept scenario (1), where the set of actual minds somehow lines up perfectly with the infinitesimally small subset of possible minds that can be simulated by Turing machines. Such an extraordinary claim would require extraordinary evidence. If we adopt scenario (1) and thus accept the claim **N**, we would have to explain exactly why a mind that's not simulatable by a Turing machine couldn't be an actual mind, even though it is a possible mind.

For similar considerations of simplicity, I also think that we should reject scenario (3). This scenario would have to explain why some actual minds cannot be simulated and some actual minds can be simulated by Turing machines. Psychologically, there isn't anything that suggests that our minds and their cognitive systems are that fundamentally different. Biology suggests that my mind operates in pretty much the same fashion as your mind, with no large structural differences between two healthy adult minds. Perhaps, in the grand scheme of our psychology, whether we can be simulated by Turing machines

is a relatively small thing that depends on chance. But this explanation doesn't seem quite too satisfactory. Without a way to explain how some actual minds are different than others, this scenario leaves us with a large explanatory gap that can't be adequately explained. It requires another explanatory layer, making it a less simple solution than the other scenarios. Thus, I think that scenario (3) doesn't provide a satisfactory answer. I don't think the mechanist would be too pleased with scenario (3) either, since it means the end of the line for his claim **N**.

This leaves scenario (2) to explain the relationship between minds and Turing machines. I think the claim that *no actual mind is Turing-simulatable* provides the best explanation out of the three. This scenario doesn't depend on any special explanation for why actual minds can't be simulated by Turing machines. If we are to defer to simplicity considerations, then we must conclude that scenario (2) is the correct one.

So to conclude, given the evidence that there exist possible minds that cannot be simulated by Turing machines, and given the evidence that the number of such minds overwhelms the population of simulatable minds, I believe that the best, simplest explanation of *actual* minds is that they simply can't be simulated by Turing machines. That is, the best explanation does not simply reject **N**—it claims outright that no actual mind can be simulated at all by a Turing machine. Any other explanation will require much more detail—detail that we do not have thus far—in order to explain how actual minds can be simulated by Turing machines.

5 Consequences

In the previous section, I demonstrated that the most reasonable explanation for actual minds is that they cannot be simulated by Turing machines, not even in the "weak" sense. Now this might be a strange conclusion because we often think of the mind's neural activity as akin to a computer processing sensory data and acting upon it. What are the

consequences of such a conclusion?

It's time to re-examine the mechanist thesis again. In Lucas's and Benacerraf's papers, they considered a very narrow definition of mechanism that only dealt with formal systems simulating the mind's arithmetic capability. If we accepted this narrow definition of mechanism instead, then we will have to defer to Lucas's and Benacerraf's original arguments. Benacerraf's argument seems like the most solid option if we are to examine this *mechanism in terms of formal systems*, and as I've previously explained, he concludes that either we are not formal systems or we cannot know that we are formal systems.

Of course, this paper is premised on the fact that the output of a Turing machine doesn't necessarily correspond to the theorems derivable in a formal system of classical first-order logic. In this paper, I examined *mechanism in terms of Turing machines* instead of formal systems. I looked specifically at simulating a mind's beliefs through the acceptor-machine or enumerator-machine models. My conclusion that a mind's beliefs are not simulated by a Turing machine certainly rules out this specific formulation of mechanism.

But maybe we should broaden our scope. The mechanist theses **M** and **N** are actually quite restrictive to the mechanist because they require a specific type of automaton—a Turing machine—for simulating the mind. So an anti-mechanist claim such as **A** that targets these definitions won't rule out any *other* formulations of mechanism that are broader in the range of machines allowed to constitute a simulation. For example, if we take mechanism as the thesis that the mind can be completely simulated by *some sort of mechanical contraption* as opposed to a *Turing machine* specifically, then mechanism can still hold. After all, I have only ruled out Turing-machine mechanism. The definition of "machine" as opposed to "Turing machine" is relatively informal, allowing for the possibility of machines that are not Turing machines. And perhaps one of these non-Turing machines can completely simulate a mind.

According to the *narrow version of the Church-Turing thesis*[18], Turing machines can simulate exactly those processes that can be described completely in terms of effectively calculable functions (Copeland). The narrow version of the thesis is quite uncontroversial and generally accepted to be true. So if a mind's beliefs and activity could not be completely described in terms of effectively calculable functions, then it would not be simulatable by a Turing machine. According to Copeland, "It is an open question whether a completed neuroscience will employ functions that are not effectively calculable". My conclusion in this paper suggests that, since the mind can't be simulated by a Turing machine, it must necessarily involve processes that can't be represented by effectively calculable functions. So a complete model of the mind would involve functions that are not effectively calculable.

But what does it mean to have a function that is not effectively calculable? There are several phenomena that can't be captured completely by effectively calculable functions. One is the concept of *uncomputable* numbers. We know that there are countably many Turing machine configurations, but uncountably many real numbers. Some real numbers (such as $\frac{2}{3}, 1, 1.5$, and π) can be computed, that is, we can come up with an effective, mechanical algorithm for printing them in some kind of binary representation. (Indeed, a simple litmus test is the fact that I can discuss these numbers in writing right here!) But there are other real numbers for which there is no such effective procedure. So we can imagine that the mind's activity might involve the storage of these uncomputable real numbers.

There is little that is undesirable about this first case. It even seems plausible that we could develop an extension of modern computers that could somehow deal with these

[18]There are multiple versions of the Church-Turing thesis. Some, like this formulation, equate the things that can be computed by Turing machines with effectively calculable functions, while others equate the capability of Turing machines with the more general class of things that can be mechanically computed. It's important to distinguish between the *narrow* and *broad* versions of the thesis. See Copeland's article "The Church-Turing Thesis" in the Stanford Encyclopedia of Philosophy for a discussion on the various versions of the thesis.

uncomputable real numbers. (A computer dealing with uncomputable numbers would be oxymoronic, indeed!)

However, there's a second obvious class of activity that can't be captured by effectively calculable functions, either. *True randomness* can't be captured completely by calculable functions, because calculable functions are deterministic on any given input[19]. After all, they are simply mechanical algorithms that transform a given input into a desired output. So we could imagine that the activity of the mind is not deterministic. Although this is perhaps less desirable than the first case, it is still conceivable. Perhaps quantum phenomena—truly random phenomena—drive mental activity in some way that cannot be simulated accurately by a deterministic computer.

These are just two obvious examples of activity that would cause the mind to not be simulatable by a Turing machine. Although we have concluded that the most reasonable explanation of the mind says that it's not simulated by a Turing machine, we still don't know *exactly what* activity in the mind causes it to be this way. Uncomputable numbers and randomness are only two possible explanations.

But even if Turing machines cannot simulate minds, this does not mean that *computers in general* cannot simulate minds. This would be the case only if we accept the *broad version of the Church-Turing thesis*[20], which gives only one view as to what computers in general can do compared to Turing machines. This broad thesis says that Turing machines are capable of any computation that can be calculated by a machine "working on finite data in accordance with a finite program of instructions" (Copeland). Compared to the narrow version of the thesis, which limits itself to effectively calculable functions, this broad version of the thesis makes a sweeping claim about computations performed on

[19]There are "nondeterministic Turing machines", but the name is a bit of a misnomer. They are provably equivalent in power to regular Turing machines, and thus the results they give do not actually depend on randomness.

[20]Copeland calls this thesis **M** to avoid confusion with the narrow version of the Church-Turing thesis, but I have unfortunately already used up the letter **M** in this paper. I call this the "broad Church-Turing thesis" because many call it, perhaps incorrectly, the "Church-Turing thesis" without any qualification.

machines in general.

Here, "machine" is rather informal and vague. It does not limit itself to any digital computer or mechanical automaton that we are capable of constructing in the present. Consider this scenario involving a possible "machine". If a neuroscientist built an atom-by-atom replica of a brain in his laboratory, and then hooked it up to a digital computer, could we count this a "machine" as opposed to just an organism? It's certainly man-made and not a naturally-occurring brain. Yet the neuroscientist's creation would certainly "simulate" the original brain from which it was copied. In fact, it seems like any mental activity in the copied brain would be *identical* to the activity in the original brain. If we concede that this exact duplicate is a mechanical computer, then we would have to reject the broad version of the Church-Turing thesis, since the mental simulation could be performed on a "machine" of some sort, even if it could not be performed on a Turing machine. Indeed, compared to the narrow version of the Church-Turing thesis, the broad version remains controversial.

In the end, my conclusion is quite simple and not as grandiose as it may seem. I don't claim that we can't simulate the mind in any mechanical way. That still seems to be an open question. What I am willing to claim is that, in the best explanation of things, *actual minds are not simulatable by Turing machines.* If we desire to simulate the mind mechanically, we must look past Turing machines as a model of the mind.

Works Cited

Alama, Jesse. "The Lambda Calculus". *The Stanford Encyclopedia of Philosophy*. Ed. Edward N. Zalta. Spring 2015. 2015. Web.

Barker-Plummer, David. "Turing Machines". *The Stanford Encyclopedia of Philosophy*. Ed. Edward N. Zalta. Summer 2013. 2013. Web.

Benacerraf, Paul. "God, the Devil, and Gödel". *The Monist* 51.1 (1967): 9–32. Web.

Boolos, George S., John P. Burgess, and Richard C. Jeffrey. *Computability and Logic*. 5th ed. Cambridge University Press, 2007. Print.

Copeland, B. Jack. "The Church-Turing Thesis". *The Stanford Encyclopedia of Philosophy*. Ed. Edward N. Zalta. Fall 2008. 2008. Web.

Hodges, Andrew. "Alan Turing". *The Stanford Encyclopedia of Philosophy*. Ed. Edward N. Zalta. Winter 2013. 2013. Web.

Lucas, J. R. "Minds, Machines, and Gödel". *Philosophy* 36.137 (1961): 112–127. Web.

Weisstein, Eric W. "Presburger Arithmetic". *Wolfram MathWorld*. Web.