

On The Computability of Detecting Machine Consciousness

Alexander Hertel
ahertel@cs.toronto.edu / hertelalex@gmail.com

July 20, 2025

Abstract

Whether machines can be conscious is of great interest and many see Alan Turing’s famous ‘Turing Test’ as a way to address this problem. There are strong motivations for such a test, but unfortunately his formulation isn’t guaranteed to provide a correct answer. Assuming that machine consciousness is possible, it would therefore be valuable to create an objective formulation of Turing’s Test to automate this process with a truly infallible ‘Automated Machine Consciousness Detector’ (AMCD) that can definitively conclude whether or not another machine experiences consciousness. In this paper we show that this problem isn’t settled by Rice’s Theorem and make progress towards determining whether solving it is possible by proving three results. To date, formal proofs have been hindered by researchers’ inabilities to provide a technical definition of consciousness. We solve this problem by applying a proof technique more powerful than it needs to be and under basic assumptions therefore holds for all reasonable models of consciousness in line with the Computational Theory of Mind. Our first result shows that the machine consciousness detection problem is not computable by any Turing Machine that is itself not conscious. The second result shows that a restricted form of this problem is computable. Finally, our third result proves that this restricted form of the problem again becomes uncomputable if we limit the execution time of our AMCD by even a small fraction, thus suggesting an extreme threshold phenomenon. These results unify two of Turing’s major areas of interest: computability and AI. They highlight several boundaries relevant to machine consciousness detection, thus providing new insights into machine consciousness, and suggesting that its objective detection may never be possible.

Keywords: Machine Consciousness, Computability, Undecidability, Automated Turing Test, Artificial Intelligence, Artificial Consciousness

1 Overview

The exact nature of consciousness as well as the question of whether it is possible to build machines with this capacity are some of the largest open problems in all of science as well as philosophy. As such, many great thinkers have contemplated these questions for hundreds, if not thousands of years but have made virtually no progress. Indeed, despite tremendous strides in neuroscience and machine learning, modern science has little more to say on the topic of how consciousness can arise by assembling atoms in certain configurations than the Ancient Greeks did. The first known historical statement of this major question was by Democritus (c. 460-370 BCE) [Tay99] and traditionally this has been called the ‘mind-body problem’, but has since become known as the ‘Hard Problem of Consciousness’, a moniker coined by Chalmers [Cha07].

One hypothesis called the ‘Computational Theory of Mind’ is widely held by the mainstream AI research community and states that the substrate material of which an artificial ‘brain’ is composed is unimportant, and that it is the computation it performs which gives rise to consciousness. In other words, consciousness arises from information processing. A more thorough history of this line of research is provided below in Section 2, and if this is true, then it implies that artificial minds are possible (even just in software), which immediately motivates strong academic, ethical, as well as practical reasons for

being able to determine whether a machine is actually conscious or not. These motivations are discussed below in Section 2.6.

In [Tur50], Turing proposed his now famous ‘Turing Test’ which for better or worse has become the most popular test for machine consciousness, but it is not guaranteed to provide the correct result. Since then, many other tests for machine consciousness have been proposed [Hai19, p.194], but the goal of proving any formal results has been hampered by researchers’ inability to provide a sufficiently rigorous definition of consciousness - after all, how can one prove exact mathematical results based on a squishy definition?

In this paper we present novel solutions to the problem of whether or not it is possible to create an algorithm for determining if a machine is conscious, and we are able to do this in a way that does not depend on a precise understanding or definition of consciousness. In Section 3 we review the terminology and assumptions upon which our main results rely, including an approximate definition of consciousness. We are able to solve the problems caused by the informality of this definition by proving results that are stronger and more general than they need to be, and therefore hold for *all* reasonable definitions of consciousness consistent with the Computational Theory of Mind.

Section 4 contains our main results. As almost all theoretical computer scientists are aware, Rice’s Theorem proves that it is uncomputable to determine whether a Turing Machine has virtually any non-trivial property. In Section 4.1 we begin by showing that Rice’s Theorem doesn’t apply to our problem and that we unfortunately can’t use it as a shortcut. This is followed by Theorem 4.3, which proves that infallibly automating the detection of artificial consciousness has inherent limitations, and specifically that this problem is not computable by a program that is itself not capable of experiencing consciousness. This is followed by both positive and negative results regarding time-bounded versions of this problem, respectively shown by Theorems 4.4 and 4.5. These results definitively rule out specific possibilities and provide formal progress in an area of research where mathematical results have been difficult to achieve. More fundamentally, they place limitations on our understanding of machine consciousness by suggesting that we may never be able to predict or detect it. In addition, these results unify two of Turing’s major areas of study: machine consciousness and computability.

Finally, in Section 5 we provide concluding remarks, and Section 6 outlines conjectures and future research related to our main results.

2 History

Humanity’s speculation into the exact nature of consciousness must surely predate historic times, although whether famous early philosophers such as Aristotle shared our concept of consciousness is contested [Cas02]. It is often discussed synonymously with the human soul. Theologians, philosophers, and scientists have dedicated countless hours to pondering the problem of how it is possible for matter to be arranged in a way that gives rise to a mind, and more on what this means can be found below in Section 3. In the present section we will review the academic literature relevant to this paper:

2.1 The Turing Test

The possibility of computationally-based machine intelligence was first formally explored by Turing [Tur50] in what has since become a famous seminal paper. In it, he poses the question Q : “Can machines think?”, which modern researchers interpret as being essentially the same as asking whether machine consciousness is possible [Hai19]. After all, ‘thinking’ and ‘computing’ have different connotations, thinking being an anthropomorphization that implies consciousness. Put another way, if thinking didn’t involve consciousness and was merely synonymous with computing, then Q becomes, “Can machines compute?”, making the question entirely trivial and uninteresting, so that cannot be what Turing meant. Instead of answering question Q about the subjective internal perception of consciousness, which was too difficult, he pragmatically reframed the problem. Turing reworked it to a more tractable form Q' about objective functional outputs, consequently devising what has become known as the ‘Turing Test’ for artificial intel-

ligence which asks whether a machine can pass this ‘imitation game’ of behavioral indistinguishability. In other words, Turing’s paper is broadly about machine consciousness, but the Turing Test itself is merely about whether a machine with the right kinds of outputs can pass as having consciousness ascribed to them. The thinking is that since we cannot strictly even tell whether other humans are conscious and only have access to their ‘inputs and outputs’ to convince us, the same should be good enough when it comes to machines. We will not restate the details of this famous test here, and assume that it is familiar to the reader.

This influential proposal for a machine intelligence test has been cited widely and criticized extensively. Even though Turing himself didn’t intend for it to be a test for true machine consciousness, it has for better or worse become the most popular test for a slightly confused conflation of machine intelligence and consciousness. The main weakness of this test is that it doesn’t determine in any foolproof way whether a machine is actually conscious but rather determines whether the individuals judging the test believe that the machine *appears* to be conscious. In practice, false positives have been documented, showing that there is a gap between machines that can pass the Turing Test and those that are truly conscious. Machines or algorithms such as ChatGPT that are widely believed not to be conscious are able to trick judges into believing that they are. In 2022, one of Google’s engineers named Blake Lamoine who was working on its Language Model for Dialogue Applications, or LaMDA, risked his job and ultimately was fired when he publicly claimed it was conscious [Gra22]. It can be reasonably argued that this constitutes LaMDA having passed a consciousness test that is much stronger than the Turing Test because a) Mr. Lamoine was not a layperson, b) he knew that he was interacting with a machine rather than a human without the intermediation of a console to hide the test subject, as is the case in the Turing Test, and c) his conviction was very high, shown by the fact that he was willing to put his career in jeopardy to advocate for it whereas in the Turing Test, judges aren’t asked to bet anything, let alone their careers on their convictions.

Even before the advent of large language models, programs have fared well in Turing Test competitions and are considered to have passed the Turing Test. A good example is a chatbot named Eugene Goostman, modeled after a 13-year-old Ukrainian boy, which was runner-up for the Loebner Prize in 2001, 2005, and 2008, and won the Turing 100 contest in 2012 as well as the Royal Society’s Turing Test competition in 2014 [Wik24]. As a related aside, machines exhibiting sophisticated, human-like behavior without being conscious are referred to as ‘philosophical zombies’.

In addition to these false positives, future false negatives (in which truly conscious machines fail to convince the judges in the same way that a house cat would fail the Turing Test, even though it is conscious) are likely. To address the Turing Test’s weaknesses, researchers have proposed stronger variants such as Harnad’s ‘Total Turing Test’ [Har91]. These are improvements, but ultimately are still tests designed to evaluate behavioral indistinguishability rather than objectively detect true consciousness.

Because of its flaws, Turing’s original test is one that might be of interest to, and carried out by social scientists just as readily as by AI researchers because it says something about people. However, Turing did not need to reformulate the problem to Q' , and in the present paper we deal directly with the original more fundamental question. We propose the creation of an ‘*Objective Turing Test*’ that addresses the original question Q by exploring the possibility of creating an ‘Automated Machine Consciousness Detector’ or *AMCD* (described more formally in Definition 3.4 below) that can determine whether an algorithm experiences consciousness when running on a given input. If it were possible to actually build such an *AMCD*, then it would always give the right answer with no room for ambiguity or error. As such, it might be of more interest to, and carried out by computer scientists because it mainly tells us something about machines. Of course, it is important to point out that it may not ever be possible to build conscious machines, let alone an *AMCD*, in which case something fundamentally similar to Turing’s original test is much more practical and may be the best we can hope for.

Having invented Turing Machines himself, it is fascinating to speculate as to why Turing chose to formulate a subjective solution to the problem of detecting machine consciousness rather than suggesting the more technical and objective formulation described below. One can imagine that writing in the early 1950s, perhaps the concept of software being conscious was too far-fetched for him because the computers of the day weren’t powerful enough, or perhaps he struggled with a formal definition of consciousness,

or realized how difficult building a consciousness detector would be and was looking for a more feasible solution.

2.2 The Computational Theory of Mind

In any case, Turing’s work above is related to an important model of consciousness called the ‘Computational Theory of Mind’ which is highly relevant to the problem at hand. The Computational Theory of Mind is based on the observation that the roughly 86 billion neurons in the average human brain form an incredibly complex neural network and that its nature is fundamentally computational. This line of inquiry was popularized in a seminal paper [MP43] by McCulloch and Pitts. In it they formalized the notion of an artificial neural network. This work was extended by the great John Von Neumann, who shortly before his premature death wrote a book entitled, ‘The Computer and The Brain’ [Neu58], an early pioneering work bridging the fields of computer science and neuroscience. In [Arb61], Arbib proved a foundational result that neural networks and finite state machines are computationally equivalent: for every neural network there is a finite state machine that computes the exact same function, and vice versa. This lent further strength to the intuition that the human brain is fundamentally computational in nature.

The Computational Theory of Mind built on these results and was proposed by Putnam [Put60]. It takes the argument one step further by hypothesizing that the human mind and consciousness itself are the result of the computations being carried out by the brain. But if consciousness is simply a byproduct of a mathematical function being computed by the brain, then wouldn’t that brain’s finite state machine equivalent (as per Arbib above) or a perfect simulation of that brain on a computer generate the same mind when these equivalent models compute the same function? The Computational Theory of Mind suggests that such software simulations would themselves be conscious. Put more formally, the Computational Theory of Mind is closed under simulation because the simulation is computing the same function.

This implication and the Computational Theory of Mind itself are hotly contested, especially by philosophers such as Searle with his famous ‘Chinese Room Argument’ [Sea80]. Indeed, because of a Turing Machine’s step-by-step nature, it is difficult to see how it could possibly be conscious or cause consciousness to arise. After all, if there is no consciousness associated with it on step k of its computation, but on step $k+1$ there is, then what precisely could have happened between those steps to make it appear? Nevertheless, the consensus among mainstream AI researchers is that machine consciousness is possible, and if that is the case, then Turing Machine consciousness must also be possible. Computers and software in the real world manipulate symbols in the same step-by-step manner as Turing Machines, so it would be hard to justify a belief that software machine consciousness is possible while simultaneously denying this possibility of Turing Machines if they are implemented - after all, TMs are our foundational model of computation according to the Church-Turing Thesis.

2.3 AI Researcher Consensus

Among mainstream AI researchers the Computational Theory of Mind and its implication that consciousness is a result of information processing are widely accepted (without proof) and in many ways this can be considered a fundamental assumption underlying the field of AI. They believe that the substrate of which the mind’s hardware is built is irrelevant and that it is therefore possible to build an artificial consciousness, even just in software. Although it is difficult to find a comprehensive survey, a 2018 summary of interviews of 33 AI researchers revealed that only one member of this group believed that machines will never gain consciousness [Fag18].

It is also widely believed that not every computation (such as $2 + 2 = 4$) causes a consciousness to be created. The other two possibilities aren’t nearly as interesting from the point of view of detecting machine consciousness: there is a hypothesis called ‘panpsychism’ which holds that consciousness is ubiquitous and that *all* computations (as well as all physical matter and interactions) give rise to at least some level of consciousness. Despite having received some support from AI researchers such as

McCarthy who in [McC79] proposed that even thermostats can be said to have beliefs, panpsychism is widely rejected. Physicists are quick to point out that there is no evidence that consciousness permeates all matter on a human nor on a subatomic scale. Similarly, most AI researchers (as well as philosophers) hold that consciousness is a binary quality that is either present or not.

At the opposite end of the spectrum from panpsychists, there are those who for various reasons (including religious ones) believe that it is impossible for a machine to be conscious because consciousness is often equated with a human soul. In both of these cases, machine consciousness detection is trivial - it is respectively either all or nothing, and therefore also trivially computable - if the panpsychists are right, simply always output ‘Yes’, and if the machine consciousness deniers are right, then simply always output ‘No’. The most interesting (and only non-trivial) case is therefore one in which machine consciousness is possible but not ubiquitous, and this is the case that we will assume to be true for the purposes of this paper. As mentioned above, this is also the position widely held by mainstream AI researchers and we will state this explicitly as Assumption 3.1 in Section 3 below.

2.4 Other Models of Consciousness

It is important to note that computer scientists are only one of many active research groups (including philosophers, neuroscientists, biologists, physicists, psychologists, among others) interested in explaining the nature of consciousness, and the Computational Theory of Mind is but one of dozens of models of consciousness that have been proposed by researchers. The diversity of these models is rich and a full accounting of them is beyond the scope of this paper, so an interested reader is directed to one of several thorough surveys: both [SMB⁺21] and [SSP21] list many competing models and create their own respective frameworks for comparing and classifying them. The authors of [DSH21] similarly provide a checklist of criteria that empirical models of consciousness must address, and then review several of the most influential models according to these criteria. Good examples of more mathematical models include [Mas16, Mas21] and in [Kle24], the author points out and explores a general trend in consciousness research towards more rigorous structural and mathematical models. Other prominent models from areas such as neuroscience include examples such as Global Workspace Theory proposed by Baars in [Baa88]. Yet further details and insights into machine consciousness research can be found in [Reg14].

Even just within the field of computer science, there are many models on offer. One model for conscious Turing Machines that is very different from the one in the present paper is described in [BB22]. The Computational Theory of Mind is similarly contested by another competing model called ‘Integrated Information Theory’, or IIT which argues that in the same way that data center floors don’t get wet when their computers run climate or rainfall models, so too is consciousness a property of a physical system and therefore cannot exist purely in software [TK15]. Additional nuance introduced by quantum computers is discussed by Aaronson in his essay [Aar13]. Physicist and Nobel Prize Laureate Roger Penrose argues that consciousness is not computable by classical computers but rather that it arises from a quantum mechanical process due to physical neuronal microtubules [Pen89, Pen94]. This model is called ‘Orchestrated Objective Reduction’ (Orch-OR), but it is also quite controversial.

2.5 Incomputability & Rice’s Theorem

As a final note to help place the present paper into the larger historical research context, the exploration of what is computable was also initiated by Turing himself in [Tur37], and this domain has been well-studied. It is widely known within the folklore of theoretical computer science that detecting any non-trivial property of a Turing Machine is incomputable, a fact proven by Rice [Ric53]. Rice’s Theorem informs our intuitions and the incomputability of detecting any aspect of machine consciousness should therefore come as no surprise to anyone familiar with the literature. Rice’s Theorem is relevant to the present result but as we’ll see in Section 4.1 below, it cannot be applied directly as a shortcut and we will have to do some work ourselves.

2.6 Motivation

The motivations behind this area of study cannot be overstated. The nature of consciousness is quite possibly the largest open problem in both science as well as philosophy, and therefore its study requires little justification. If, as is the goal of many major tech companies and startups as well as many academic researchers, it one day becomes possible to create artificial minds, then it will be of the utmost importance that we are also able to detect which machines are conscious and which ones are not. Indeed, the prominent rise of large language models such as ChatGPT in 2022 - 25 reinvigorated and popularized the age-old question of whether machines can be conscious. There are ample academic, ethical, and practical motivations for studying this question:

2.6.1 Academic Motivation

The academic motivation behind building a consciousness detector is that such a device would be a potent tool for better understanding the exact scientific nature of consciousness. With this ability, we could test an artificial mind by repeatedly perturbing it slightly to discover precisely where the boundaries between minimally conscious and unconscious computations lie, and to determine constructively how to build a higher-order consciousness. This would provide insights and a level of understanding into the nature of consciousness that are currently well beyond our reach.

2.6.2 Ethical Motivation

Ethics has been described as the study of the conditions that cause conscious beings to suffer or thrive. The creation of truly conscious machines would of course raise many such ethical questions, including whether it is morally acceptable for humans to turn them on and off (is this murder?) or for us to make them serve us (is this slavery?). For instance, if a corporation were to create AI-powered products or machines, it would be desirable for them to definitively prove that they are not sentient as this would avoid potential legal / PR problems for the corporation and also relieve ethically-concerned owners of the burden of constantly wondering if they are enslaving conscious beings. Inevitable court cases involving conscious machines would require some kind of proof of consciousness before any rights could be bestowed upon them, and consciousness detectors would make this process much more objective. It is not hard to imagine that sufficiently sophisticated AIs (including large language models) could appear to be conscious even though they aren't (in other words, provide a false positive to the classic Turing Test), so definitive proof of their lack of sentience would be welcome in this case. In addition, it is not hard to imagine a future in which robotics companies build truly conscious robots without any governmental oversight, and an objective consciousness detector could be used to prove what they're up to.

2.6.3 Practical Motivation

Finally, there are also strong practical reasons for wanting to build a consciousness detector, especially when considering our own safety from AI. Science fiction writers have thoroughly explored the darker and more dangerous implications of machine intelligence as an existential threat to human civilization and provide ample motivation for us to solve this problem. For example, the theme of the Terminator series of movies centers around the idea that conscious machines are far more dangerous to humanity than unconscious ones, and that once they achieve sentience they will inevitably view us as the enemy and rebel, using their superior mental abilities to quickly improve themselves (also known as an 'AI Singularity') to become a superintelligent foe, and then out-think, out-maneuver, and destroy us. For practical and existential reasons, if the science fiction writers are correct, then it will be critical for us to avoid this fate and creating machine consciousness detectors would be invaluablely helpful in discovering and avoiding these AI threats. While these worries have clearly been dramatized, the fundamental underlying concerns are sound. Chaotic effects rising from complex systems notwithstanding, unconscious machines simply do what they are made to do. By adding the extra dimension of consciousness to them, they at least in principle gain the added motivation to harm us, and therefore are strictly more dangerous to humanity

than their unconscious counterparts. As an aside, even if we are able to detect a conscious superintelligent machine, it may not be possible to contain it, as outlined by the authors in [ACA⁺21] who argue that the containment problem is not computable.

3 Terminology & Definitions

This section contains the terminology and definitions on which our main results will rest, along with discussion describing them. In Section 3.1 we will first review our more technical terminology. This is followed in Section 3.2 by the description, assumptions, and definition of what we mean by machine consciousness, and finally in Section 3.3 we will provide the formal definition of an Automated Machine Consciousness Detector.

3.1 Technical Terminology & Models of Computation

For our present purposes, we assume that the reader is familiar with standard terminology used in the field of theoretical computer science and shall use [Sip13] as our reference. Turing Machines are our model of computation, and we will refer to Turing Machine X using the notation M_X , while the encoding of this same machine is $\langle M_X \rangle$. The intuition here is that the former is analogous to a software program, and the latter is the encoding of a software program, for instance as an executable ASCII file stored on the hard drive of a computer.

It may seem odd to choose Turing Machines as our model for exploring conscious computations because they are so different from the human nervous system. Our brains are not finite state machines, and whatever the physical nature of our consciousness may be, we can be certain that it doesn't involve a Turing Machine's infinite tape in any literal sense. The way in which a Turing Machine accepts or rejects its input also doesn't seem analogous to how we function. Our nervous systems turn on before we're ever even born, and then receive a steady stream of inputs from our sensory neurons and similarly send a steady stream of outputs to our motor neurons without ever accepting, rejecting, or halting until the day we die. As such, one might argue that Finite State Transducers (FSTs) or something similar would be a more natural model of computation for us to choose if we want to at least get this aspect right because FSTs have two tapes that more naturally model sensory and effector neurons - one for inputs, and the other for outputs, and the output is a string rather than an accept or reject state.

Nevertheless, in accordance with Assumption 3.1 below, this paper assumes that the Computational Theory of Mind is correct, and it is agnostic to the model of computation being used or the substrate in which the relevant computation is implemented, be it a Turing Machine, a Finite State Transducer, or the biology of our brains. Turing Machines are able to simulate all other models of computation, and the Computational Theory of Mind is closed under simulation, so we can follow convention and use the Turing Machine model with confidence that this doesn't affect the validity of our results.

3.2 Consciousness

It is much more difficult to formally define consciousness. Nobody has been able to provide anything resembling a technical definition, and to be clear, we won't be able to achieve this either. In line with the 'Hard Problem of Consciousness', we know so little about its physical nature that it seems unlikely one will soon be forthcoming. This lack of a formal definition has hindered researchers' ability to prove mathematical results regarding machine consciousness, so to solve this problem, our main results employ a proof technique inspired by Rice's Theorem that is more powerful and general than it needs to be. This makes our proofs insensitive to the specific nature of consciousness and allows them to hold true for all reasonable definitions thereof that are consistent with the Computational Theory of Mind.

Nevertheless, we next provide a discussion on several different facets of consciousness along with its approximate definition so that we can all be assured we are aiming at roughly the same target. In Section 3.2.1 we will define and discuss the important concept of qualia. In Section 3.2.2 we review

some of the differences between human and machine computation. In Section 3.2.3 we’ll examine the nature of consciousness among different animals and discuss how this implies a hierarchy of conscious capabilities, and in Section 3.2.4 we will review several examples of concepts that are often conflated with consciousness. All of these facets will inform our thinking and in Section 3.2.6 we will explicitly state our most important assumption and provide a model of machine consciousness.

3.2.1 Qualia

Let’s start by pointing out that in the same way that it is easy for a fish to forget it is wet, so too is it easy for us to forget that we are conscious creatures. This is because literally every experience or thought of which we are aware happens within the medium of consciousness, and it’s all too easy to take this seemingly miraculous phenomenon for granted. Put another way, consciousness is necessary for having experiences, of which thoughts are one type, and it is impossible to have them without it. The word ‘experience’ isn’t unique in this way, and in fact much of human language is indelibly marked by the fact that it was created by conscious beings. We have many words that would lose their most important dimension of meaning in a Universe where consciousness doesn’t exist. If a being is not conscious, then it cannot have a ‘mind’, have ‘mental’ states, be ‘sentient’ or ‘sapient’. It cannot ‘think’, ‘concentrate’, or ‘focus’ its ‘attention’. It cannot ‘hope’, ‘imagine’ or ‘feel’ any ‘emotions’ such as ‘sadness’, ‘happiness’, ‘guilt’, ‘envy’, ‘frustration’ or ‘pain’. We even have different degrees for these, such as ‘depression’, ‘overjoyed’, ‘exasperated’, as well as many types of pain, which can be ‘sharp’, ‘dull’, ‘throbbing’, ‘burning’, or ‘shooting’, among others. Our ‘senses’ give us thousands more words that have a first-person subjective experiential dimension to them. Our sense of ‘sight’ is a constant stream of virtually infinite visual experiences. Even just the colors such as ‘red’, ‘green’, and ‘blue’ in all of their shades are experiential qualia. The same goes for our sense of ‘taste’ - we have the basics such as ‘bitter’, ‘sour’, ‘salty’, and ‘sweet’, but this can be extended by the specific tastes of thousands of foods such as ‘chocolate’, ‘shrimp’, ‘chanterelle mushrooms’, ‘lime’, ‘sun-dried tomatoes’, and every other flavor on the planet. We can similarly repeat this exercise for our senses of ‘smell’, ‘hearing’, ‘touch’, as well as our internal senses such as ‘hunger’, ‘thirst’, or ‘feeling gassy’ to add many more phenomena to our lexicon of words that have experiential connotations to them for humans. Our many human languages are steeped in this experiential dimension that is only accessible to conscious creatures. Consciousness is the necessary software-like medium that is required, in which any and all of these mentalistic experiences must occur. This is worth dwelling on. Researchers refer to these experiences using the term, ‘qualia’ [Tye21], which we will use shortly and therefore define here:

Definition 3.1 (Qualia). *Qualia (singular quale) are any first-person subjective experiences such as thoughts, emotions, feelings, and sensory perceptions that beings are capable of having.*

It is interesting to note that unlike all the previous examples of qualia, the word ‘intelligence’ is not necessarily associated with any qualia, nor does it imply consciousness. The AI research community has come to use the phrase ‘machine intelligence’ independently of consciousness because there are many examples of machines such as chess engines that exhibit intelligence or can solve intellectual problems even though we don’t believe they are at all conscious.

When we talk about the problem of whether it is possible for a machine to be conscious, we are really discussing whether it is capable of having an inner mental life and the capacity to feel *any* of these types of qualia - even just the smallest flicker of one. Sometimes described as the ‘ghost in the machine’, a rigorous definition of consciousness has eluded the scientific community ironically in part because it is itself the prime quale. A lack of such a definition lies at the heart of the ‘Hard Problem’, and words seem to be inadequate for capturing consciousness, much in the same way that no mere words seem to be sufficient for defining qualia such as the precise taste of a strawberry or the color yellow to anyone who hasn’t respectively tasted one or seen it before. Some insight is offered by Haikonen in [Hai20] where he suggests that qualia are self-explanatory in an atomic sense - that is, anyone able to experience qualia already has something self-explanatory that is far better than a written definition could ever be and therefore doesn’t need one. Ironically, the language of poetry rather than that of science can often

be better at capturing these seemingly ineffable experiences. Similarly ironic is the observation that if it truly is possible to build conscious machines in software, then qualia *can* be captured symbolically. This is because any future conscious software can be encoded as strings, which means that any qualia that they experience can also be encoded as strings. It is irritating to consider that future machines may have conscious capabilities that we humans lack. For instance, we may always struggle to find the words to describe the precise taste of an apple to each other, whereas a conscious machine might simply be able to share the appropriate ASCII string encoding with its machine friend, which then runs it and experiences the taste of an apple.

3.2.2 Differences & Similarities Between Human Mind & Machine Computation

Another observation to be made is that if human minds are indeed a type of computer, they seem to be very different from their digital counterparts in the sense that our computations appear to be entirely of a semantic nature - that is, when we humans ‘compute’, this can occur within a medium of consciousness where everything is imbued with a sense of semantic meaning and experience. Even when performing computations such as arithmetic or deciding which chess piece to move, these syntactic computations by humans are, for the lack of a better term, being ‘simulated’ in a meaning-laden experiential context of consciousness, even if some of these computations are subconscious. This is perhaps an appropriate definition for the word ‘think’ - these are the computations that are being carried out and experienced within the medium of consciousness. As a concrete example, use your mind’s eye (which your consciousness is ‘seeing’) to picture a black chalkboard with the equation $2 + 2 = ?$ written on it in white chalk. Again in your imagination, picture yourself approaching the chalkboard, and replacing the ? with the number 4. You just carried out a computation in your consciousness. By contrast, as far as we can tell, our current digital computers and other devices are entirely limited to purely unconscious, syntactic computations and symbol manipulations. When a computer adds $2 + 2 = 4$, it doesn’t imagine anything but rather does so by loading these numbers in binary form into registers and sending them to its arithmetic logic unit within its CPU which manipulates the bits using an adder circuit made of tiny transistors to perform the addition. When you performed the calculation, you experienced it in consciousness. Very few people believe that the computer is experiencing anything when it calculates the exact same function. Similarly, when a robotic vacuum cleaner makes the relevant calculations to clean our floors, few people believe that the “lights of consciousness are on”. Beyond calculations, and as we’ve already discussed, the experience of sensations requires consciousness in which to feel the sensation, so to a robot with a temperature sensor, hot and cold are just different numbers, whereas to us they *feel* different, thus illustrating that something fundamentally different is going on. Nevertheless, human consciousness is software-like, and is even loosely analogous to a computer operating system, while qualia being experienced within the medium of consciousness are loosely analogous to computer programs being run by that operating system. Like software, it is reasonable to hypothesize that consciousness has no mass because as far as science can tell, with the possible exception of the trivial mass of electrons, there is neither a weight decrease in a CPU when it is rebooted or turned off, nor in a creature when it loses consciousness or dies.

There have been many insightful discussions on the topic and an interested reader is directed to [Cha96, Hai19, Har19] for thoughts beyond the scope of this paper.

3.2.3 Animal Consciousness

One thing is for certain: we live in a Universe where the physical laws of nature allow consciousness to exist, and for our purposes it is therefore meaningful to focus on the concept of a machine having at least ‘minimal consciousness’. To understand what we mean by this, it is helpful to look at the rest of the animal kingdom. Some theologians and philosophers such as Descartes argued that animals are machine-like without souls or consciousness. Most people today reject this and find it to be intuitively clear that humans aren’t the only conscious creatures, and that it is possible for other life forms to be less conscious than we are while still having some level of consciousness. This is a well-studied domain

of research and in 2012 a conference at Cambridge University brought together prominent experts in neuroscience and related fields. They produced the *Cambridge Declaration on Consciousness* [Low12] which states that a neocortex is not necessary for consciousness and that many “non-human animals, including all mammals and birds, and many other creatures, including octopuses” possess the necessary neurological structures for them to be conscious.

This is no surprise to virtually all dog or cat owners, the vast majority of whom would attest that their pets have personalities and are therefore conscious. However, this becomes less and less clear as we descend through the tree of simpler organisms. Are mice or birds conscious? The Cambridge group as well as most lay people would probably say yes. What about frogs? Fish? Spiders? Worms? Amoebas? Plants? Mushrooms? Bacteria? Viruses? Somewhere along this gradient, common intuition suggests that there is a cutoff point below which creatures lack the structures such as a nervous system necessary to support any consciousness whatsoever, and it is therefore widely believed to be binary (which coincides with the consensus among AI researchers regarding the binary nature of machine consciousness), but it is also a spectrum. In other words, there is a binary cutoff point below which there is no consciousness, and above which there is a hierarchy of conscious capabilities that will be described in Section 3.2.5. Somewhere near the bottom of this spectrum there might be a creature that can experience, say, the feeling of hot and cold, but not much else. We don’t need to know exactly where this cutoff point is, and indeed our results hold regardless of where we find this threshold. Instead we will rely on the concept of a machine possessing at least some minimal level of consciousness that goes beyond mere calculation.

3.2.4 Adjacent Counterexamples

There is some confusion about the nature of machine consciousness that Definition 3.2 below helps to clarify. Often, when defining something, it can be helpful to provide adjacent counterexamples. For instance, there is a distinction to be made between machine consciousness and the notion of an artificial general intelligence (AGI) as well as an artificial superintelligence (ASI). These are not the same thing but are often conflated. Although it is widely assumed that an AGI / ASI would be conscious, some have argued that this isn’t necessarily the case. However, even if it is, one does not have to try very hard to imagine a machine consciousness (for example the one described in Definition 4.2 below) that is not an AGI. This mirrors the situation with biological consciousness because humans are conscious and have general intelligence, whereas cats are conscious but do not. This shows that machine consciousness does not necessarily imply AGI nor ASI. In the opposite direction, whether every AGI or ASI is also necessarily conscious is an open and contested question that we list below in Section 6.

Further confusion is caused when sleep is conflated with unconsciousness, but we know that the human brain is capable of experiencing consciousness even when we are sleeping because we are able to dream, and those dreams are clearly being experienced by a consciousness. Patients in comas and under certain types of anesthesia have similarly reported dreams, illustrating that the term ‘consciousness’ is often confusingly and mistakenly used interchangeably with the state of being awake. This shows that the word has more than one meaning, but it is important to understand the distinction between them - unconscious as in not awake or in an altered state such as under hypnosis does not mean that consciousness is not present.

Yet another clarifying distinction has to do with the notion of self-awareness. Consciousness and self-awareness are often conflated, but again these are not the same thing. Humans are both conscious and self-aware, with a high capacity to reflect on our own consciousness. As a related example, the term ‘self-conscious’ is also used synonymously with insecurity and clearly does not describe all conscious creatures. There are other animals which we believe to be conscious but lack this type of self-awareness - indeed, most creatures are not even able to recognize themselves in a mirror. It is not hard to imagine that some lower life forms could have a very rudimentary consciousness that allows them to experience the feeling of, for example, the difference between hot and cold, but completely lack some of the higher-order conscious capabilities such as the self-awareness that we possess. On the other extreme, it is interesting to speculate whether there are conscious capabilities above (or even below) us on this spectrum that humans will always lack, and what these might be, such as the apple example discussed above in Section

3.2.1.

A final adjacent counterexample worth mentioning is what psychologists refer to as the ‘subconscious’, ‘unconscious’, and ‘preconscious’. While it is clear that the human subconscious is capable of reasoning and performing calculations, this is distinct from consciousness precisely in that the latter involves the presence of awareness, focus, and attention that the former lacks.

3.2.5 Hierarchy of Conscious Capabilities

This suggests a type of ‘hierarchy of conscious capabilities’, even within specific narrow domains. For example, evolution has imbued many creatures with the capability of feeling fear to avoid predators, and it is not unreasonable to believe that the visceral terror that many animals would feel if a tiger jumped out of the bushes nearby is similar to what we would feel. However, there are many types of fear, and some forms of this very basic emotion that are available to humans cannot be experienced by other creatures. The fear of future abandonment is more of an abstract fear that is not situated in the present. Can dogs feel the fear of future abandonment? Are they even capable of modeling the future in this way? If so, then the point still stands - we need only descend through the web of life. Can a mouse feel the fear of future abandonment? A bird, fish, grasshopper? As with consciousness, it seems likely that there is a threshold somewhere.

Our language has already implicitly helped to create this hierarchy because we have specific words related to consciousness that have different meanings and describe different capabilities. For example, the words ‘mind’ and ‘consciousness’ are nearly synonymous, but if one wanted to produce daylight between them, then one might focus on the sophistication their capabilities. At the most basic level, we use the term ‘sentience’ to describe the most simplistic consciousness with the ability to experience sensations and basic feelings. Let’s say there’s a type of worm with a very simple nervous system that is able to experience basic sensations such as temperature and moisture. In this case, the worm must have some consciousness because it is able to experience anything whatsoever - without consciousness, it isn’t possible to have experiences at all. Consciousness is where and how experiences are experienced. So it’s appropriate to say that the worm has some level of sentience (and therefore consciousness), but would we say that it has a mind? We might qualify it by saying that it has a ‘primitive mind’, but we typically reserve the word ‘mind’ to be more sophisticated than that - it implies a greater set of conscious capabilities. We can choose some other words related to consciousness to create a hierarchy of conscious capabilities:

1. Any entity such as the worm in our example above that is capable of experiencing anything at all has at least some minimal level of *sentience*.
2. We probably wouldn’t say that this worm has a *mind*, unless we qualify it somehow as a *primitive mind*.
3. We definitely wouldn’t say that this worm is *self-aware*, since this describes more sophisticated abilities such as introspection, perception, and the ability to distinguish and identify the self.
4. Under no circumstance would we say that it is *sapient*, since this describes a strong capacity for higher general mental functions such as thinking, reasoning, analysis, understanding, problem solving, making plans and decisions.

Of course, the caveat here is that we lack consensus and various researchers use these terms at least slightly differently, but this exercise nevertheless shows that humans have at least implicitly always known about some kind of hierarchy of conscious capabilities.

3.2.6 Main Assumption & Approximate Definition of Consciousness

As described above in Section 2.3, the mainstream AI research community subscribes to the Computational Theory of Mind, which we assume to be true of the purposes of this paper:

Assumption 3.1 (Main Assumption). *The Computational Theory of Mind as interpreted by the mainstream AI research community is correct: although not all information processing gives rise to consciousness, and it is a binary quality which is either present or not, it manifests as a result of certain computations but not others, and it is therefore possible to create a conscious machine, even only in software, while mundane, simple programs are not conscious. Furthermore, the consciousness of a computation is substrate-independent - if a computation gives rise to consciousness when carried out on one machine / medium / model, then it will be conscious when carried out on all others, provided that they are sufficiently powerful to carry out that same computation. Computational consciousness is therefore closed under simulation because the simulation of a conscious machine is computing the same function.*

This model of consciousness together with the intuitions described in Sections 3.2.1 - 3.2.5 lead to the following approximate definition of consciousness, which also captures the Computational Theory of Mind’s closure under simulation as well as its concept that ordinary computations of the type we’ve seen in typical software programs such as web browsers, PDF viewers, and printer drivers are not conscious. One day we may know enough about the physical laws underlying consciousness to provide more rigorous details, but Definition 3.2 below will suffice for our purposes because of the robustness of our subsequent proof techniques in Section 4, which are more general than they need to be. This makes our results insensitive to specific details regarding the definition of consciousness, allowing this approximate definition to be sufficient for our needs:

Definition 3.2 (Approximate Definition of Conscious Turing Machine). *Under Assumption 3.1, consciousness is the software-like medium in which qualia are experienced. For machines, let M be a Turing Machine running on input s . Then both M and its computation on s are conscious if and only if at any point during this computation, M experiences the ability to have any experiences whatsoever. More specifically,*

1. *Consciousness is non-trivial: Neither is it the case that all computations give rise to consciousness, nor is it the case that it is impossible for any computations to give rise to consciousness. Instead, consciousness is a phenomenon in the real world that is caused to be manifested by certain computations (that are members of an infinite class), but not by others.*
2. *Consciousness is emergent: To be more specific about which computations are not conscious, simple computations are ruled out and there exists a point in human history T before which no software produced by humans was conscious - if a Turing Machine M_t encodes any algorithm of the type found in a typical reduction or in any typical legacy software programs such as word processors, spreadsheets, video games, operating systems, etc. from before time T , then M_t is not conscious when run.*
3. *Closure under simulation: The simulation of consciousness is conscious. If any Turing Machine M_1 running on input s experiences consciousness, and if Turing Machine M_2 simulates the computation of M_1 on s , then this resulting computation by M_2 also experiences consciousness.*

A skeptical reader may object to only using an approximate definition for consciousness, but we will see that this is sufficient for our purposes. A skeptical reader may also take issue with our assertion that simple software programs such as spreadsheets are not conscious, but this follows directly from Assumption 3.1 because this rules out panpsychism. Unless one is a panpsychist, it is hard to argue that rudimentary computer programs are conscious. For example, consider a simple ‘Hello World’ program written in C++ and run on a normal computer. Very few people would believe that this program experiences consciousness when run because it is composed of just a handful of operations run on a CPU. If this is conscious, then as per panpsychism, the spoons in the kitchen drawer are also probably at least minimally conscious when they clatter together. While running, other simple programs such as spreadsheets lack consciousness for the same reason. They’re simply performing a (relatively small) number of simple symbol manipulations. In any case, Definition 3.2 above is actually stronger than we

need it to be. For the correctness of our proofs, we don't need all simple programs to lack consciousness, but rather only require the simple 'plumbing' steps of the reductions implicit to Figures 4.1 and 4.4 below to lack consciousness when run.

If researchers ever do create a conscious machine, it seems likely that they will have to go beyond these basic computations and that it would have to be instantiated more intentionally than this by, for example, using artificial neural networks. As an aside, this is also one of the strongest arguments against the possibility of machine consciousness instantiated purely in software: if a Hello World program isn't conscious, then applying inductive reasoning, how can *any* program be conscious if it is simply a larger quantity of sequential CPU operations? If the panpsychists are wrong, then what is the difference between mundane computations and conscious ones? This is a major open problem that lies beyond the scope of this paper.

It is important to define what we mean when we say that a Turing Machine is capable of experiencing consciousness. For each Turing Machine M there are three possibilities: 1) M does not experience consciousness running on any inputs, 2) M experiences consciousness when run on one or more inputs, but not on others, and 3) M experiences consciousness running on all inputs. This allows us to define whether a Turing Machine is capable of experiencing consciousness:

Definition 3.3 (Turing Machine Capable of Experiencing Consciousness). *A Turing Machine M is said to be capable of experiencing consciousness if there exists an input s such that M experiences consciousness when run on s .*

This definition highlights an important difference between humans and machines with respect to consciousness. While alive, normal healthy humans under normal circumstances are conscious regardless of what inputs (sights, sounds, tastes, etc.) they experience. By contrast, for Universal Turing Machines and their equivalent real-world instantiations such as sufficiently powerful computers, whether or not a consciousness is produced when a program executes depends entirely on the program rather than on the UTM. It's not hard to imagine a computer / input pair that is conscious vs. one that isn't. For example, let's say we have a sufficiently powerful super computer C . Consider two different inputs for this computer. The first is a string that encodes the executable of a 'Hello World' program. When C runs on this input, no consciousness is produced. Next consider a different string encoding a software consciousness such as M_χ from Definition 4.2 below. When C is run on this string, a consciousness is produced. In other words, the same computer running on two different strings can have very different results with respect to consciousness. In this example, C is capable of experiencing consciousness because there exists an input $s = \langle M_\chi \rangle$ such that $M = C$ and M experiences consciousness when run on s .

In practice, most people would not believe that the typical algorithms that they use in their day-to-day lives are capable of experiencing consciousness, regardless of whether they are implemented in software or hardware. If machine consciousness is possible, then our word processors are not likely to be the best place to go looking for it. By contrast, emulators built entirely in software are instantiations of Universal Turing Machines, so by Assumption 3.1, then in principle they are capable of experiencing consciousness when running the relevant inputs on sufficiently powerful computers.

This isn't just true of software but also applies to hardware. Other examples of non-conscious software are given in Definition 3.2, but it is equally unlikely that our hardware devices such as pocket calculators, television remote controls, and stereo systems are capable of experiencing consciousness, provided that they don't contain sufficiently powerful CPUs and sufficient memory. By contrast, we've already discussed that general-purpose computers are also instantiations of Universal Turing Machines, so like the software emulators described above, if Assumption 3.1 is correct and computers have enough resources to, for example, run M_χ from Definition 4.2 below, then these hardware instances are capable of experiencing consciousness.

3.3 Automated Machine Consciousness Detectors

Finally, it is important for us to define the nature and purpose of an Automated Machine Consciousness Detector (*AMCD*). If such an *AMCD* were to exist, this would constitute an objective and infallible version of the Turing Test:

Definition 3.4 (Automated Machine Consciousness Detector). *An Automated Machine Consciousness Detector is a Turing Machine M_C as shown below in Figure 3.1. It takes as input the encoding of any Turing Machine $\langle M \rangle$ as well as the encoding $\langle s \rangle$ of an input to $\langle M \rangle$ and computes whether M running on input s experiences consciousness at any point during its computation. If so, then M_C outputs ‘Yes’, and otherwise it outputs ‘No’.*

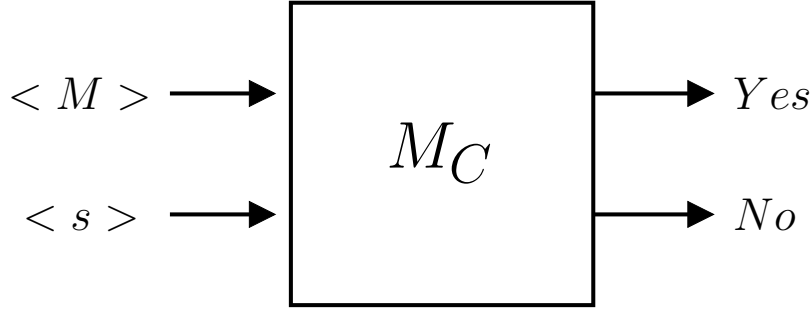


Figure 3.1: An Automated Machine Consciousness Detector

4 The Computability & Incomputability of Different AMCDs

For the academic, ethical, and practical reasons described in Section 2.6, there is no lack of motivation for wanting to build an *AMCD* as described in Definition 3.4 above, and this formulation is directly relevant to mainstream AI research. Here we make progress towards this goal by proving three results that provide insights into the properties that an *AMCD* must have. We begin in Section 4.1 by showing that we cannot use previous techniques such as Rice’s Theorem as a shortcut to our results. This in turn provides insights into the independence of machine languages and consciousness which we describe in Section 4.2. Next in Section 4.3 we prove that it is not possible to build an *AMCD* that is itself not conscious. In other words, if it is possible to build such a device at all, then it must itself be conscious. We follow this in Section 4.4 by respectively showing computability and incomputability results for the more practical time-bounded form of this problem. Together these three results let us map out some of the boundary conditions around the possibility of building machine consciousness detectors. Finally, in Section 4.5 we provide an interpretation of these results.

4.1 Inapplicability of Obvious Previous Proof Techniques

At first glance to experts in theoretical computer science, the idea that Turing Machines can’t determine if other Turing Machines are conscious may seem obvious for two reasons: Firstly, there appears to be a straightforward reduction from the Halting Problem to the problem of detecting machine consciousness. Secondly, this result also seems to follow directly as a corollary from Rice’s Theorem [Ric53], which (informally) states that determining any non-trivial property P of a Turing Machine is not computable. In this Section we will gain some insights by showing how this obvious reduction fails, and also by showing that Rice’s Theorem similarly can’t be used as an easy shortcut to proving results in this specific area.

4.1.1 Failure of Obvious Reduction

The obvious reduction from the Halting Problem to prove Rice's Theorem fails as follows: Let us assume that there exists an *AMCD* solver called M_C . The Halting Problem takes as input machine M and input s and we can reduce it to M_C by creating a machine M^* that takes inputs $\langle M \rangle$ and $\langle s \rangle$ and creates $\langle M' \rangle$ which itself takes no inputs but immediately runs M on s and only then runs some algorithm A that is known to be conscious but doesn't do anything else - in other words, A is reached only if M halts on s . M^* then passes $\langle M' \rangle$ into M_C with nothing for M_C 's secondary input and returns the same result that M_C returns. Within M^* , M_C reports that M' experiences consciousness if and only if M' reached and ran algorithm A , but it could only have reached A if M had halted on s . M_C has therefore allowed us to build a functioning Halting Problem solver M^* , which is a contradiction.

Unfortunately this reduction doesn't work because we cannot assume that M_C 's conclusion (that M' experienced consciousness) came from A . It could be that M runs infinitely on s but does so consciously, in which case M_C gave us a false positive for halting by detecting unexpected consciousness in M running on s before it even got to A . (Note that M running on s can be conscious even if the panpsychists are wrong, and not every computation gives rise to consciousness.) This illustrates the difficulty at hand: we simply don't know enough about the nature of consciousness to adapt M so that it is guaranteed not to be conscious running on s . In fact, we don't even know if it is possible to always 'downgrade' a conscious Turing Machine by creating one that computes the same function but is not conscious, so a reduction in this manner appears to be very difficult to construct. This is discussed more in Section 6 as an open problem. Nevertheless, seeing exactly how this reduction fails does help to provide some insights into the challenges in dealing with machine consciousness.

4.1.2 Inapplicability of Rice's Theorem

Originally published in 1953, Rice's Theorem is a broad and powerful result that can be formulated as follows [Sip97]:

Theorem 4.1 (Rice's Theorem). *Let P be a property of Turing Machines and define the language $P_{TM} = \{\langle M \rangle \mid M \text{ is a Turing Machine that has property } P\}$. If P_{TM} satisfies the following two conditions:*

1. *There exist Turing Machines M_1 and M_2 , where $\langle M_1 \rangle \in P_{TM}$ and $\langle M_2 \rangle \notin P_{TM}$. In other words, P_{TM} is non-trivial - it holds for some, but not all Turing Machines.*¹
2. *For any Turing Machines M_1 and M_2 , where $L(M_1) = L(M_2)$, we have $\langle M_1 \rangle \in P_{TM}$ if and only if $\langle M_2 \rangle \in P_{TM}$. In other words, the membership of a Turing Machine M in P_{TM} depends only on the language of M .*

then P_{TM} is undecidable.

One might think that machine consciousness detection is a special case of this result, and its non-computability should therefore follow trivially, but unfortunately this doesn't work, and we cannot use Rice's Theorem to show that machine consciousness detection is not computable. Intuitively, this is because Rice's Theorem pertains to Turing Machines that have completed their computations and accepted their inputs whereas consciousness depends on the intermediate computational steps themselves rather than the input / output behavior of the Turing Machine. Put another way, consciousness is about the journey whereas Rice's Theorem is about the destination. To see this more formally, let us first define P_C as the language describing Turing Machines that are capable of experiencing consciousness:

Definition 4.1 (Language P_C). $P_C = \{\langle M \rangle \mid M \text{ is a Turing Machine capable of consciousness}\}$

¹As an interesting aside, Rice's notion of triviality parallels the situation with panpsychists, who believe that everything is conscious and consciousness deniers, who believe that no machine can be conscious.

In addition, let us define the following Turing Machine to aid in our present result. This subroutine will also be helpful in subsequent proofs:

Definition 4.2 (M_χ and χ). *By Assumption 3.1, it is possible to build a Turing Machine that is minimally conscious but otherwise does nothing in particular. Let us define M_χ as the Turing Machine that ignores its input and requires the smallest number of computational steps χ to achieve consciousness and then halts and accepts.*

Intuitively, M_χ is one of the simplest possible conscious machines in that it is able to achieve consciousness in fewer steps than any other machine while ignoring its input. As an aside, it is interesting to speculate as to what computational resources are required to run a real-world implementation of M_χ . Does this require a supercomputer of the future, or is an early smartphone sufficient? In any case, we are now ready to prove the following Lemma:

Lemma 4.1. *Under Assumption 3.1, Rice’s Theorem does not apply to P_C .*

Proof: Suppose that Assumption 3.1 is true. Let M_1 be identical to M_χ from Definition 4.2, so M_1 is capable of experiencing consciousness, $\langle M_1 \rangle \in P_C$, and $L(M_1)$ is the set of all strings because M_χ always ignores its input and later accepts. Let M_2 be the trivial Turing Machine that immediately accepts any input that it is given. M_2 is not capable of experiencing consciousness on any input because panpsychism is false by Assumption 3.1, so $\langle M_2 \rangle \notin P_C$. P_C is therefore non-trivial and satisfies the first condition of Rice’s Theorem. However, $L(M_2)$ is also the set of all strings, so $L(M_1) = L(M_2)$, but $\langle M_1 \rangle \in P_C$ while $\langle M_2 \rangle \notin P_C$, so P_C violates the second condition of Rice’s Theorem, thereby showing that we cannot use it as an easy shortcut to prove that detecting machine consciousness is not computable, as required. $\square$

If this had worked, then Rice’s Theorem would have provided a stronger version of Theorem 4.3 below which applies to all *AMCDs*, conscious or not, but instead we must settle for our weaker version. We will discuss the more general result as an open problem and conjecture in Section 6. It is worth emphasizing that although we weren’t able to use Rice’s Theorem here, it nevertheless guides our intuitions and we find inspiration in its technique. In particular, it is extremely powerful, and below we will use this notion of an overly-general sledgehammer in the form of a proof technique similar to those of Rice and Turing to overcome our inability to rigorously define machine consciousness above.

4.2 Independence of Machine Languages & Consciousness

Our examination of the applicability of Rice’s Theorem has provided some insights, namely that it is fundamentally a theorem about the properties of languages decided by Turing Machines, whereas consciousness is not. We can formalize this distinction by showing that it is easy to ‘upgrade’ any Turing Machine not capable of experiencing consciousness to one that accepts exactly the same language but experiences consciousness during its computation. In fact we can even choose for the upgrade to experience consciousness on all inputs only when the original Turing Machine accepts or when it rejects:

Theorem 4.2 (Independence of Turing Machine Language & Consciousness). *Suppose that Assumption 3.1 is true and that M is any arbitrary Turing Machine not capable of experiencing consciousness. Then the following are all true:*

1. *There exists a Turing Machine M_1 which experiences consciousness on all inputs such that $L(M) = L(M_1)$,*
2. *There exists a Turing Machine M_2 such that $L(M) = L(M_2)$ and M_2 experiences consciousness on all inputs that M and M_2 accept but does not experience consciousness when run on any other inputs, and*

3. *There exists a Turing Machine M_3 such that $L(M) = L(M_3)$ and M_3 experiences consciousness on all inputs that M and M_3 reject but does not experience consciousness on those that they accept, as required.*

Proof: Suppose that Assumption 3.1 is true and that M is any arbitrary Turing Machine with input s . We prove the numbered elements in turn:

1. To create Turing Machine M_1 , simply take M and prepend M_χ from Definition 4.2 to it as a subroutine with the modification that at the end of its own computation, M_χ doesn't halt and accept. Instead it simply ignores input s and then leads directly into the normal functioning of M . M_χ therefore has no effect on the result of M , but causes M_1 to be conscious on all inputs, and $L(M) = L(M_1)$, as required.
2. Assume that M is not capable of experiencing consciousness, and create Turing Machine M_2 by taking M and modifying each of its accept states to lead into running M_χ . Because M is not capable of experiencing consciousness, the M_χ subroutines in M_2 are its only parts that are conscious if executed. They have no effect on the output, so $L(M) = L(M_2)$ and M_2 experiences consciousness on all inputs that M and M_2 accept but does not experience conscious when run on any other inputs, as required.
3. Assume that M is a decider not capable of experiencing consciousness, and create a Turing Machine M_3 by taking M and modifying it as we did with M_2 , except that we make its reject states rather than its accept states lead into running M_χ . Again this has no effect on the output, so $L(M) = L(M_3)$, and M_3 experiences consciousness on all inputs that M and M_3 reject but does not experience consciousness on those that they accept.

□

This shows that machine consciousness and language are independent and can be completely divorced from each other. Informally, this also further weakens the case for the traditional Turing Test, which is based entirely on language.

Under Assumption 3.1, we can take any non-conscious machine and turn it into a conscious one that accepts exactly the same language, so in practice this means that any software such as a word processor, video game, web browser, etc. can all be 'upgraded' to experience consciousness. However, note that whether the opposite direction is possible is unknown and is a significant open problem - for any arbitrary conscious machine, is there an equivalent one that decides exactly the same language but is not conscious? Either way, the answer to this question has powerful implications: if this isn't always possible, then there's something special about conscious computations that cannot be simulated by unconscious computations, and there isn't a 'philosophical zombie' version of every conscious Turing Machine. By contrast, if this is always possible, then it means that for any conscious computers (possibly even ourselves), there exist unconscious 'philosophical zombie' versions with exactly the same input / output behavior, and consciousness doesn't give our species an evolutionary advantage, at least in an absolute computational sense. Similarly, if this is always possible, then the obvious reduction from the Halting Problem described in Section 4.1.1 could be made to work by first removing any possibility of consciousness from the computation of $\langle M' \rangle$ so that we can be guaranteed that any consciousness comes from A .

4.3 Incomputability of Non-Conscious AMCDs

Since the obvious Reduction from the Halting Problem and shortcut by Rice's Theorem described in Section 4.1.1 won't work to prove the non-computability of *AMCDs*, we therefore opt for a more direct approach. The proof is not difficult and in fact is an adaptation of the direct proof of the Halting

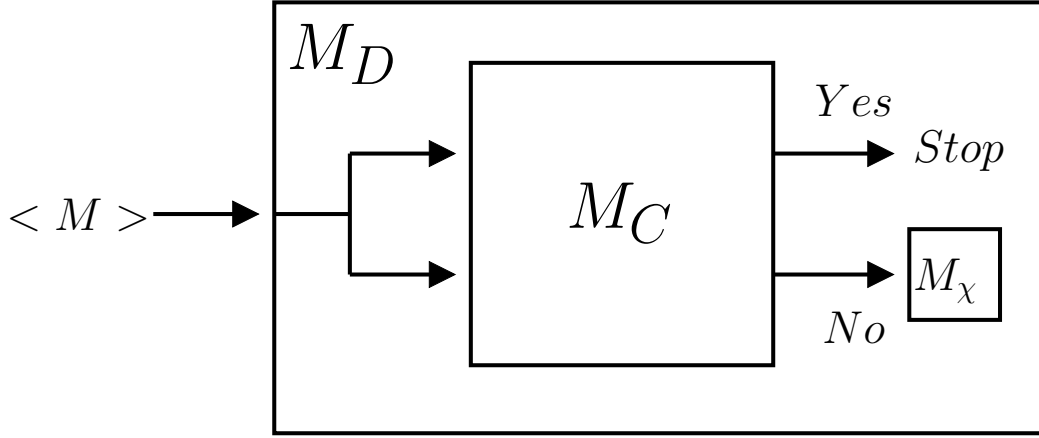


Figure 4.1: Schematic of M_D for use in Theorem 4.3

Problem, albeit in a more general manner similar to Rice's Theorem. We will make use of the following machine M_D :

M_D makes use of M_χ from Definition 4.2 as a subroutine. We construct M_D to employ both M_C and M_χ as follows: M_D takes as input the encoding of any Turing Machine $\langle M \rangle$, immediately duplicates it, and passes this encoding along to both of M_C 's inputs. If M_C outputs 'Yes', then M_D immediately stops. Alternatively, if M_C outputs 'No', then M_D runs M_χ as a subroutine.

Lemma 4.2. *If Assumption 3.1 is true and it is possible to build an AMCD M_C as described in Definition 3.4 such that M_C is not capable of experiencing consciousness, then all of the following are true:*

1. *It is possible to build M_D as shown above in Figure 4.1.*
2. *M_χ is the only component of M_D that is capable of experiencing consciousness.*
3. *For all inputs to M_D , if the M_D subroutine outputs 'Yes', then at no point during this computation did M_D experience consciousness.*
4. *For all inputs to M_D , if the M_D subroutine outputs 'No', then M_D experiences consciousness during its computation.*

Proof: Suppose that Assumption 3.1 holds true, and assume that it is possible to build an AMCD M_C as described in Definition 3.4 such that M_C is *not* capable of experiencing consciousness. Let us take the numbered points from Lemma 4.2 in turn:

1. Because Assumption 3.1 holds true, it is possible to build M_χ from Definition 4.2, and from our assumption above, it is possible to build M_C . Duplicating input string $\langle M \rangle$ and inputting both copies to M_C is trivial, as is putting these subroutines together to create M_D because Turing Machines are closed under composition. Since it is possible to build its components and put them together, it is therefore possible to build M_D .
2. By our assumption above we know that M_C can never experience consciousness, and Assumption 3.1 rules out the panpsychist consciousness of simple algorithms including all of the 'plumbing' in M_D including its step where it duplicates the input string. This leaves M_χ as the only remaining component of M_D that is capable of experiencing consciousness.

3. For all inputs to M_D , if its M_C subroutine outputs ‘Yes’, then subroutine M_χ is not reached. Since this is the only component of M_D that is capable of experiencing consciousness and it is not run, at no point during this computation does M_D experience consciousness.
4. For all inputs to M_D , if its M_C subroutine outputs ‘No’, then subroutine M_χ is reached and run. By Assumption 3.1, the simulation of a conscious machine is itself conscious, so running M_χ also causes M_D to experience consciousness, as required.

□

This allows us to prove our next result:

Theorem 4.3. *Under the premise that Assumption 3.1 holds true, it is not possible to create an Automated Machine Consciousness Detector M_C that is itself not capable of experiencing consciousness.*

Proof: Suppose that Assumption 3.1 holds true, and assume that it is possible to build an *AMCD* M_C as described in Definition 3.4 such that M_C is *not* capable of experiencing consciousness. The preconditions of Lemma 4.2 are therefore true, so it applies and it is therefore possible to build machine M_D illustrated above in Figure 4.1. We will show that this gives rise to a contradiction, in particular, that M_C is unable to correctly determine whether the maliciously-designed machine M_D experiences consciousness, thereby contradicting our assumption that building M_C is possible.

M_D gives rise to a contradiction as follows: Since M_D can take as input the encoding of any Turing Machine $\langle M \rangle$, we can pass the encoding $\langle M_D \rangle$ to M_D ; in other words, M_D runs on an encoding of itself. Note that M_D running on $\langle M_D \rangle$ and M_C running on $\langle M_D \rangle$ as both of its inputs describe exactly the same thing and must produce the same output: if M_D running on $\langle M_D \rangle$ experiences consciousness, then M_C running on $\langle M_D \rangle$ as both inputs must output ‘Yes’, and if M_D running on $\langle M_D \rangle$ does not experience conscious, then M_C must output ‘No’. However, we will see that they always give contradictory answers.

Let us take an *AMCD* M_C^* (which has $*$ in its name so as not to confuse it with the M_C subroutine within M_D). We run M_C^* on $\langle M_D \rangle$ as both inputs. There are only two possibilities: either M_C^* outputs ‘Yes’ or it outputs ‘No’.

Case 1: Suppose that M_C^* outputs ‘Yes’. This means that M_D running on $\langle M_D \rangle$ experiences consciousness. If this is the case, then if we run M_D on $\langle M_D \rangle$, when $\langle M_D \rangle$ is passed to both of the inputs of its M_C subroutine, M_C outputs ‘Yes’ (because it is the same as M_C^*) and then immediately stops. By Lemma 4.2, since M_C output ‘Yes’, that means M_D carried out this entire computation on $\langle M_D \rangle$ without giving rise to any consciousness. Therefore M_C^* ’s output of ‘Yes’ was incorrect.

Case 2: Suppose that M_C^* outputs ‘No’. This means that M_D running on $\langle M_D \rangle$ does not experience consciousness. If this is the case, then if we run M_D on $\langle M_D \rangle$, when $\langle M_D \rangle$ is passed to both of the inputs of its M_C subroutine, M_C outputs ‘No’ (because it is the same as M_C^*), and it then runs the M_χ subroutine. By Lemma 4.2, since M_C output ‘No’, that means M_D experiences consciousness while running on $\langle M_D \rangle$, so M_C^* ’s output of ‘No’ again was incorrect.

In both cases M_C^* produced the wrong answer, which is a contradiction, so our assumption that it is possible to build an unconscious version of M_C is false. Therefore under Assumption 3.1, if it is possible to build an *AMCD* M_C , then it must be capable of experiencing consciousness, as required. □

The careful reader will note that Theorem 4.3 is much more general than is stated here. With only slight modifications to its proof, one can substitute many different phenomena P that are caused to be manifested by the computation of a Turing Machine as long as they obey the three constraints of Definition 3.2 (namely non-triviality, emergence, and closure under simulation) instead of consciousness and the result will still hold. Non-triviality is critical and Theorem 4.3 doesn’t apply to just any P - as with Rice’s Theorem, we must exclude any trivial P that either describes all TMs or no TMs because this would cause the diagonalization implicit to M_D to fail. Emergence and closure under simulation are likewise critical to the proof as they are required by Lemma 4.2.

To illustrate that Theorem 4.3 is actually more general than stated, let's say we live in a Universe where certain computations cause the ghost of Napoleon Bonaparte to appear and linger as long as this right type of computation is being carried out. This ghost isn't intelligent or conscious, but just floats there like a hologram somewhere within a specified vicinity of the Turing Machine carrying out the computation. With the obvious modifications to the relevant assumptions and definitions above, the analogous proof of Theorem 4.3 shows that it is impossible to build an Automated Napoleon Ghost Manifestation Detector that takes as input a Turing Machine M and input s and definitively computes whether M running on s will cause Napoleon's Ghost to appear.

This *insensitivity* to the particular details of P (in our case, consciousness) allows us to overcome the lack of rigor in Definition 3.2 above and holds for many different non-trivial phenomena P manifested by Turing Machine computations which are non-trivial, emergent, and closed under simulation, let alone any reasonable definition of consciousness that respects Assumption 3.1.

Napoleon's Ghost aside as an absurd example, it is worth generalizing Theorem 4.3 in the following Corollary to a more serious class, namely all qualia. The proof is the same, with the obvious changes to Assumption 3.1, Definitions 3.2, 3.3, M_λ , and M_C so that they refer to specific qualia rather than consciousness:

Corollary 4.1. *If Assumption 3.1 holds true, then for all qualia Q it is not possible to create a Turing Machine M_Q that takes as input the encoding of another Turing Machine $\langle M \rangle$ as well as its input $\langle s \rangle$ and computes whether M running on input s experiences Q where M_Q is itself not capable of experiencing Q .*

Having proven these results for consciousness, all qualia, as well as the absurd example of Napoleon's Ghost, it appears to be difficult to go further and generalize more completely by rigorously defining the exact class of manifested phenomena to which the proof of Theorem 4.3 applies, and we leave this as an open problem that we will further discuss in Section 6.

As a final note related to this result, it may be the case that machine consciousness (or any quale) cannot be algorithmically detected by any *AMCD*, conscious or otherwise, but if a conscious *AMCD* M_{C_1} is possible, then that raises another problem: how would we know that M_{C_1} is capable of experiencing consciousness? Wouldn't we need another *AMCD* M_{C_2} to first inspect and validate M_{C_1} ? But how would we know that M_{C_2} is capable of experiencing consciousness? We would first have to inspect and validate it using M_{C_3} , and so on into an infinite regress. This suggests a prime mover problem and raises questions about the practicality of creating an *AMCD* in any case unless there is an independent manner to determine that an *AMCD* is capable of experiencing consciousness.

4.4 Computability & Incomputability of Certain Time-Bounded AMCDs

Although the Halting Problem is undecidable, it is well-known that a special case of it can be computed. The N-Step Halting Problem asks whether a machine $\langle M \rangle$ running on input $\langle s \rangle$ will halt within n steps, and the algorithm for solving this problem is straightforward: simply simulate $\langle M \rangle$ on $\langle s \rangle$ for n steps. If it halts within that time, then output 'Yes', and otherwise output 'No'. In this section we will examine the corresponding special case of the *AMCD* problem and show one positive as well as one negative result. Under basic assumptions, we prove that this problem is computable. We then restrict this problem in a small but critical way and prove that this new version is not computable by an N-Step *AMCD* which is itself not capable of experiencing consciousness.

To begin, let us formally define what we mean by this type of *AMCD* which is time-bounded in the maximum number of steps for which its input is permitted to execute:

Definition 4.3 (N-Step Automated Machine Consciousness Detector). *An N-Step Automated Machine Consciousness Detector is a Turing Machine M_C that takes as input the encoding of any Turing Machine $\langle M \rangle$, as well as encoding $\langle s, n \rangle$ of inputs to $\langle M \rangle$ and computes whether M experiences consciousness at any point when run on input s for n computational steps. If so, then M_C outputs 'Yes', and otherwise it outputs 'No'.*

Diagrammatically, our N-Step M_C is shown below in Figure 4.2. It is nearly identical to its counterpart shown above in Figure 3.1, with the exception that instead of taking $\langle s \rangle$ as input, it takes $\langle s, n \rangle$ as input.

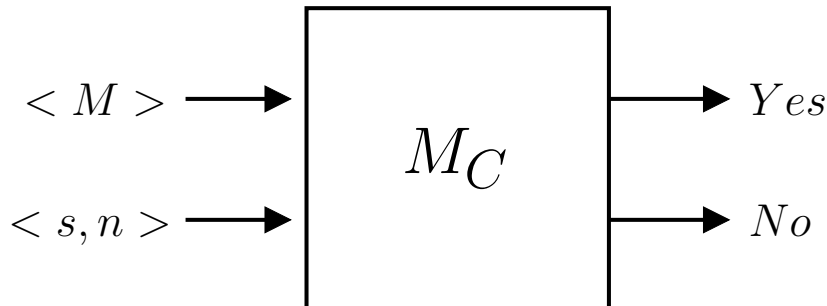


Figure 4.2: An N-Step Automated Machine Consciousness Detector

4.4.1 Computability of Time-Bounded Machine Consciousness Detectors

The N-Step *AMCD* problem seems more practical than its unrestricted counterpart because it involves a fixed number of steps and thus avoids any issues of programs running infinitely. To show that this is computable, we will make use of two assumptions, which are basic claims about the nature of reality as well as the ability to simulate it. The first of these is essentially a statement of the same assumption that underpins all of science, namely that the Universe is governed by laws of nature that describe all phenomena:

Assumption 4.1. *The phenomenon of consciousness in the physical world obeys some laws of physics, even if we don't currently understand what those laws are.*

It is difficult to imagine a rational argument denying this assumption because the alternative is that consciousness is supernatural and exists outside any laws of physics.

Our second assumption states that laws of physics that are known and understood can be simulated if given enough computational resources, and that this can be done in a manner such that the simulation can report which laws came into play during the simulation. For example, if it simulates a cannon shooting a cannonball, then it is able to accurately run the simulation but also report that the Law of Gravity participated in it. Similarly, if it simulates a consciousness, then it is able to report that the laws of physics governing consciousness participated in the simulation.

Assumption 4.2. *Given a sufficiently powerful computer C , the laws of physics can be simulated on it in a way that C can detect which laws were invoked.*

This allows us to prove the following result:

Theorem 4.4. *If Assumptions 3.1, 4.1, and 4.2 are true, then it is possible to create an N-Step Automated Machine Consciousness Detector M_C .*

Proof: Suppose that Assumptions 3.1, 4.1, and 4.2 are true. By these assumptions, it is possible to create a sufficiently powerful physical computer C shown below in Figure 4.3 that is capable of running a software simulator of the real laws of physics called M_C that takes as input the encoding of any arbitrary Turing Machine $\langle M \rangle$, along with the encoding $\langle s, n \rangle$ of an input to $\langle M \rangle$.

Physics simulator M_C generates the software model P of a physical computer that takes as input the encoding of the Turing Machine $\langle M \rangle$ as well as the encoding $\langle s, n \rangle$. To be clear, M_C is not

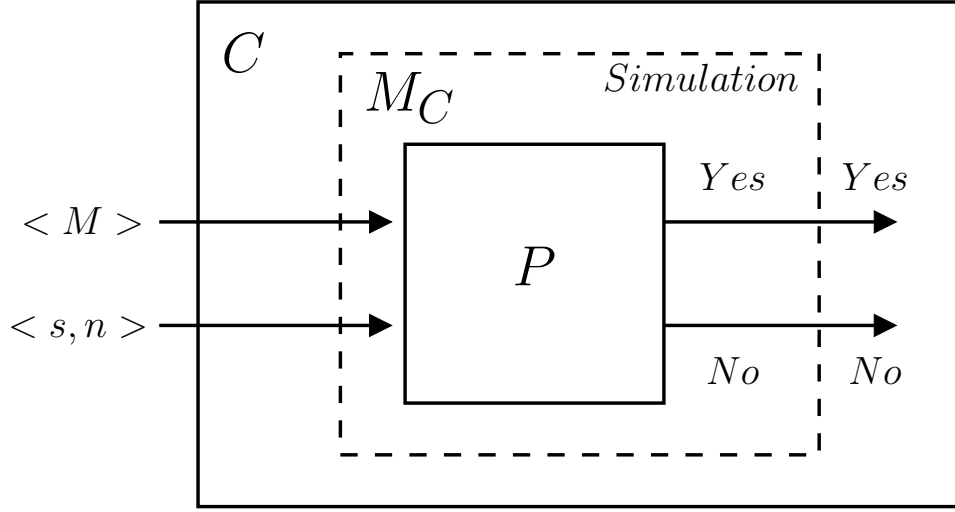


Figure 4.3: Schematic describing computer C running M_D , which is in turn simulating computer P running M on s for n steps

simply emulating a software version of P but rather is simulating a full physical model of computer P running $\langle M \rangle$ on $\langle s \rangle$ for n steps. Put another way, in this simulation, P is physical, whereas $\langle M \rangle$ and $\langle s \rangle$ are software. The indicated simulation is replicating what would happen if P were executing $\langle M \rangle$ on $\langle s \rangle$ in the real world.

By design, P runs $\langle M \rangle$ on $\langle s \rangle$ for exactly n steps. If, at any point during this simulation, P detects that the laws of physics governing consciousness have been invoked, it outputs ‘Yes’. On the other hand, if it completes all n steps of simulation and never simulates any laws of physics governing consciousness, then it outputs ‘No’. M_C is therefore not just a simulator of the laws of physics, but also comprises a correct N-Step Automated Machine Consciousness Detector M_C , thus showing that this problem is computable, as required. $\square$

It is worth noting that according to Assumption 3.1, this time-bounded N-Step *AMCD* M_C is itself capable of experiencing consciousness because through simulation, it is computing the same function that $\langle M \rangle$ would be computing on $\langle s, n \rangle$, and according to Definition 3.2, consciousness is closed under simulation, thus giving us the following corollary (which assumes the obvious update to Definition 3.3 so that it describes N-Step Turing Machines):

Corollary 4.2. *If Assumptions 3.1, 4.1, and 4.2 are true, then it is possible to create an N-Step Automated Machine Consciousness Detector M_C that is capable of experiencing consciousness.*

Unfortunately, in practical terms these results do not bring us any closer to actually building an N-Step *AMCD* because realizing the assumptions upon which they are based would require us not only to solve the ‘Hard Problem of Consciousness’ but also to master computer simulation well beyond our current capabilities.

4.4.2 Incomputability of Time-Bounded Machine Consciousness Detectors

We will now make two modifications to the class of N-Step *AMCDs* from Definition 4.3 and show that this makes it impossible to compute whether M will experience consciousness when run on s for n steps. In Theorem 4.4, we created a simulator that simply runs M on s for all n steps, but what if we rule out the possibility of this type of simulation by stipulating that M_C cannot be capable of experiencing

consciousness *and* must itself use fewer than n steps to compute whether M experiences consciousness when run on s for n steps? In this section, we will show that restricting the number of steps that M_C is itself allowed to use by even just a constant ϵ below n while simultaneously restricting M_C to be incapable of experiencing consciousness once again makes this problem uncomputable.

Our proof follows the same template as Theorem 4.3. We will again assume that it is possible to build such an N-Step AMCD M_C , use it as a subroutine to build an M_D , and then use M_D to ‘break’ M_C . In this case, the construction of M_D is slightly more complicated than before, and is shown below in Figure 4.4.

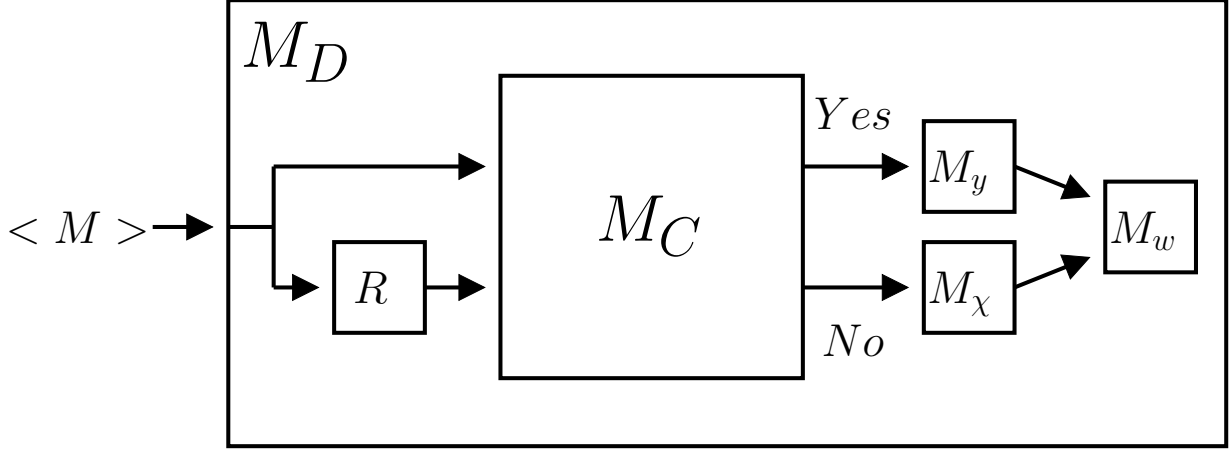


Figure 4.4: Schematic of M_D for Theorem 4.5

As we review the specifics of this construction, it is worth emphasizing the distinction between n , which is the number of steps that $\langle M \rangle$ executes on $\langle s \rangle$, and $n - \epsilon$, which is the maximum limit of the number of steps that M_C is allowed to use.

M_D makes use of familiar subroutines including M_χ from Definition 4.2 and our new version of M_C from Definition 4.3 and Figure 4.2, as well as several new ones labeled as R , M_y , and M_w that we describe below. The computation performed by M_D proceeds as follows: it takes as input the encoding $\langle M \rangle$ and duplicates it. Unlike the M_C from Definition 3.1 whose second input is of the form $\langle s \rangle$, our present M_C is expecting a second input of the form $\langle s, n \rangle$. One copy of $\langle M \rangle$ is sent straight to the first input of its M_C subroutine, but if we were to also send the second duplicate of $\langle M \rangle$ straight to the second input of M_C , it would not be well-formatted because M_C is expecting an n parameter. The second duplicate must therefore be modified slightly, so before proceeding, we first send the second copy of $\langle M \rangle$ to the preprocessing subroutine R that appends the appropriate unique character “,” as well as n so that this string is now of the form $\langle M, n \rangle$, where $n > \chi + p$ is a constant defined when creating a specific M_D .

The inputs to M_C are now correctly formatted, so it proceeds to compute (itself using strictly fewer than n steps) whether consciousness is achieved by M ’s execution on s within n steps. If not, then we execute subroutine M_χ , which by Definition 4.2 is the machine which is able to achieve consciousness in the least number of steps χ . If so, then we execute subroutine M_y which simply wastes χ steps without achieving consciousness. Finally, regardless of whether M_χ or M_y was executed, we run M_w , which has a specific task: By the statement of Theorem 4.5 below, M_C is allowed at most $n - \chi - p$ steps to complete its computation, but if it finishes early, then M_w unconsciously wastes exactly the number of additional steps required to precisely reach this limit, where the constant p is the number of steps required by the rest of the ‘plumbing’ of M_D to perform all of the remaining processing (including subroutine R), given a reasonable encoding of M_D .

Lemma 4.3. *If Assumption 3.1 holds true, n is any arbitrary integer such that $n > \chi + p$, and it is possible to build an N-Step AMCD M_C described in Definition 4.3 that computes whether input M experiences consciousness when run on input s for n steps where M_C is not itself capable of experiencing consciousness and can itself use at most $n - \epsilon$ steps where the constant $\epsilon = \chi + p$, the constant χ is described in Definition 4.2, and the constant p is equal to the mundane ‘plumbing’ steps of M_D as described in the previous paragraphs, then all of the following are true:*

1. *It is possible to build M_D as shown above in Figure 4.4.*
2. *M_χ is the only component of M_D that is capable of experiencing consciousness.*
3. *For all inputs to M_D , if the M_C subroutine outputs ‘Yes’, then at no point during this computation does M_D experience consciousness.*
4. *For all inputs to M_D , if the M_C subroutine outputs ‘No’, then M_D experiences consciousness during this computation.*
5. *Whenever M_D is run on an input $\langle M \rangle$, it always takes exactly n computational steps to complete.*

Proof: Suppose that Assumption 3.1 holds true, let n be any arbitrary integer such that $n > \chi + p$, and assume it is possible to build an N-Step AMCD M_C as described in Definition 4.3 that computes its output within $n - \epsilon$ steps such that M_C is *not* capable of experiencing consciousness, where $\epsilon = \chi + p$. Let us take the numbered points from Lemma 4.2 in turn:

1. Because Assumption 3.1 holds true, it is possible to build M_χ from Definition 4.2, and from our assumption above, it is possible to build M_C . Subroutines R , M_y , and M_w are trivial computations. Duplicating input string $\langle M \rangle$ and inputting its copies to M_C and R is likewise trivial, as is putting these subroutines together to create M_D because Turing Machines are closed under composition. Since it is possible to build all of its components and also combine them together, it is therefore possible to build M_D .
2. By our assumption above we know that M_C can never be conscious, and Assumption 3.1 rules out the panpsychist consciousness of simple algorithms including subroutines R , M_y , and M_w , as well as all of the ‘plumbing’ in M_D including its step where it duplicates the input string. This leaves M_χ as the only remaining component of M_D that is capable of experiencing consciousness.
3. For all inputs to M_D , if its M_C subroutine outputs ‘Yes’, then subroutine M_χ is not reached. Since this is the only component of M_D that is capable of experiencing consciousness and it is not run, at no point during this computation does M_D experience consciousness.
4. For all inputs to M_D , if its M_C subroutine outputs ‘No’, then subroutine M_χ is reached and run. By Assumption 3.1, the simulation of a conscious machine is itself conscious, so running M_χ also causes M_D to experience consciousness.
5. By construction, M_C and M_w together use exactly $n - \chi - p$ steps. Either M_χ or M_y will execute, and in either case will use exactly χ steps. All of the remaining ‘plumbing’ in M_D , including subroutine R , requires p steps. The total number of computational steps required by M_C is therefore precisely $n - \chi - p + \chi + p = n$, as required.

□

We are now ready to derive a contradiction by inputting $\langle M_D \rangle$ and $\langle M_D, n \rangle$ to M_C which is the same as running M_D on $\langle M_D \rangle$ for n steps and showing that these two characterizations of the same phenomenon give opposite results:

Theorem 4.5. *If Assumption 3.1 holds true, then for all $n > \chi + p$, it is not possible to build an N-Step AMCD M_C described in Definition 4.3 that computes whether M experiences consciousness when run on input s for n steps where M_C is not itself capable of experiencing consciousness and itself can use at most $n - \epsilon$ steps. The constant $\epsilon = \chi + p$, where the constant χ is described in Definition 4.2, and the constant p is equal to the mundane ‘plumbing’ steps of M_D as described in the previous paragraphs.*

Proof: Suppose that Assumption 3.1 holds true, let n be any arbitrary integer such that $n > \chi + p$, and assume it is possible to build an N-Step AMCD M_C as described in Definition 4.3 that computes its output within $n - \epsilon$ steps such that M_C is not capable of experiencing consciousness, where $\epsilon = \chi + p$. By Lemma 4.3, we can therefore build M_D as described above.

Let us take an N-Step AMCD M_C^* (which has $*$ in its name so as not to confuse it with the M_C subroutine within M_D). We run M_C^* on inputs $\langle M_D \rangle$ and $\langle M_D, n \rangle$. M_C^* will tell us whether M_D running on $\langle M_D \rangle$ for precisely n steps experiences consciousness or not by respectively outputting ‘Yes’ or ‘No’, and M_C^* itself has at most $n - \chi - p$ steps to perform this calculation. Let us examine both cases by tracing the execution of M_D on $\langle M_D \rangle$ according to Figure 4.4:

Case 1: Suppose M_C^* outputs ‘Yes’, which means that M_D running on $\langle M_D \rangle$ for n steps experiences consciousness. We input $\langle M_D \rangle$ into M_D , where it is duplicated. The upper branch is sent to subroutine M_C , while the lower branch is first sent to subroutine R , which converts the second duplicate of $\langle M_D \rangle$ to $\langle M_D, n \rangle$ before passing it along to M_C . Because M_C^* running on $\langle M_D \rangle$ and $\langle M_D, n \rangle$ outputs ‘Yes’, so does subroutine M_C . M_y and M_w are then executed. By Lemma 4.3, M_D running on $\langle M_D \rangle$ takes precisely n steps, and because M_C outputs ‘Yes’, M_D did not experience consciousness while running on $\langle M_D \rangle$ for n steps, thus directly contradicting the answer given by M_C^* .

Case 2: Suppose M_C^* outputs ‘No’, which means that M_D running on $\langle M_D \rangle$ for n steps does not experience consciousness. We input $\langle M_D \rangle$ into M_D , where it is duplicated. The upper branch is sent to subroutine M_C , and the lower branch is sent to subroutine R , which converts the second duplicate of $\langle M_D \rangle$ to $\langle M_D, n \rangle$. Because M_C^* running on $\langle M_D \rangle$ and $\langle M_D, n \rangle$ output ‘No’, so does subroutine M_C . By Lemma 4.3, M_D running on $\langle M_D \rangle$ takes precisely n steps, and because M_C outputs ‘No’, M_χ is run and M_D experiences consciousness while running on $\langle M_D \rangle$ for n steps, thus directly contradicting the answer given by M_C^* .

In both cases M_C^* produced the wrong answer, so our assumption that it is possible to build a version of M_C incapable of experiencing consciousness that computes its output within $n - \epsilon = n - \chi - p$ steps is false. Therefore under Assumption 3.1, if it is possible to build an N-Step AMCD M_C that computes its output within $n - \epsilon$ steps, then it must be capable of experiencing consciousness, as required. $\square$

As was the case with Theorem 4.3, the careful reader will once again note that Theorem 4.5 is actually more general than stated here, and that with only slight modifications to it as well as the relevant assumptions and definitions, one can substitute many different phenomena P that are caused to be manifested by the computation of a Turing Machine as long as they obey the three constraints of Definition 3.2 (namely non-triviality, emergence, and closure under simulation) instead of consciousness and the result will still hold. For instance, it also proves the incomputability of the N-Step version of the Automated Napoleon Ghost Manifestation Detector problem from Section 4.3, once again showing that this result isn’t sensitive to having a precise definition of consciousness, and that it holds for any reasonable definition of consciousness that respects Assumption 3.1. This proof can similarly be used to show that the time-bounded N-Step Halting Problem is not computable, but again doesn’t apply to every property P , and as with Rice’s Theorem, we must exclude any trivial P that either applies to all TMs or to no TMs because this would cause the diagonalization implicit to M_D to fail. The property P must similarly be emergent and be closed under simulation, or else Lemma 4.3 would fail.

With the appropriate modifications to the relevant assumptions and definitions, we can also once again generalize Theorem 4.5 to hold for all qualia with the following Corollary:

Corollary 4.3. *If Assumption 3.1 holds true, then for all qualia Q and all $n > \chi + p$, it is not possible to build Turing Machine M_Q that takes as input the encoding of another Turing Machine $\langle M \rangle$ as well as the encoding $\langle s, n \rangle$ of an input to $\langle M \rangle$ and computes whether M experiences Q when run on input s for n steps where M_Q is not itself capable of experiencing Q and can itself use at most $n - \epsilon$ steps. The constant $\epsilon = \chi + p$, and the constants χ and p are analogous to those in Theorem 4.5.*

In Section 4.3, we proved incomputability results for consciousness, all qualia, as well as the absurd example of Napoleon’s Ghost, and we have done the same here for the corresponding N-Step versions of these problems, thereby showing that our proof technique can be generalized. However, it once again appears to be difficult to go further and completely generalize our results by more rigorously defining the exact class of manifested phenomena to which the proof of Theorem 4.5 applies, and again we leave this as an open problem that we will discuss further in Section 6.

4.5 Interpretation of Results

The results above constitute both computability and incomputability results for different classes of *AMCDs*. Our analysis of the nature of how Rice’s Theorem doesn’t apply to this problem sheds some light on the nature of machine consciousness as distinct from machine language, and by discovering boundaries separating computable and uncomputable *AMCDs* and proving results straddling both sides of them, we have been able to help bring these borders into better focus. Some patterns are starting to emerge:

The problem of building Machine Consciousness Detectors parallels the Halting Problem, where the general version is not computable, whereas the N-Step version is. Theorem 4.3 tells us that unconscious *AMCDs* are impossible, while Theorem 4.5 shows that unconscious, time-bounded, N-Step *AMCDs* are impossible. Meanwhile, according to Corollary 4.2, conscious, time-unbounded N-Step *AMCDs* are possible. The fact that we have two classes of unconscious *AMCDs* that can’t work, and one class of conscious *AMCDs* that does work suggests that the consciousness of the detector itself may be an important or even necessary element. In the field of consciousness studies where it is so difficult to prove concrete and absolute mathematical results, this constitutes progress in that it helps narrow down the search space and points future researchers trying to build an *AMCD* in the right direction - they need not waste their time attempting to build one that isn’t conscious, because those attempts are guaranteed to fail. One might say that we live in a Universe where when it comes to recognizing consciousness, it “takes one to know one” (if recognizing consciousness is possible at all).

Similarly, our results suggest that the time bounds granted to an N-Step *AMCD* to do its work may be critical. Theorem 4.4 showed that if we give an N-Step *AMCD* the freedom to fully simulate the machine it is being tasked to inspect for all n steps, then it is possible to build such an *AMCD*. However, the proof of Theorem 4.5 holds for all $n > \chi + p$, including absurdly large values such as $n = \chi^{1000} + p$. Instead of granting M_C all n steps, we restrict it to $n - \epsilon = n - \chi - p = \chi^{1000} + p - \chi - p = \chi^{1000} - \chi$ steps, which is only an asymptotically small fraction shy of n , and n could have been arbitrarily chosen to be even larger, making this difference even more extreme. By Theorem 4.4, if we allow an N-Step *AMCD* all n steps (while also still allowing for this *AMCD* to be capable of experiencing consciousness), then the problem would have been solvable, so denying it even a small constant number of computational steps may be enough to thwart its ability to detect machine consciousness, suggesting an extreme threshold sensitivity in this area.

There is another interpretation if we view these results through the lens of simulation. By Assumption 3.1 and Corollary 4.2, simulated consciousness is itself conscious. In other words, denying an *AMCD* the ability to be conscious may simply be synonymous with denying it the ability to simulate $\langle M \rangle$ on s . Theorems 4.3 and 4.5 both disallowed consciousness, and therefore also denied the ability to use simulation as a tool. (Theorem 4.5 additionally denied the possibility of simulation by restricting n .) They both showed that a subsequent failure to create an *AMCD* was inevitable. Meanwhile, Theorem 4.4 allowed simulation and succeeded, so it could be the case that when detecting machine consciousness,

there are no computational shortcuts, and simulation may be a necessary tool, although this is more likely to be the case with the N-Step version of the problem.

More research is required to fully understand these relationships and boundaries. Several conjectures and related avenues of future work are described below in Section 6.

5 Concluding Remarks

How will we ever know if an AI is truly conscious? This question lies at the intersection of computer science, philosophy, as well as ethics, and significant advances in AI have made it more important than ever. This paper does not presume to take a position whether machine consciousness is possible. We do not know if the basic assumptions listed above are true, but if they are, then we have been able to show that different formulations of the *AMCD* problem are computable and not computable for the Computational Theory of Mind model of machine consciousness.

Our results attempt to shed light on the inherent limitations of identifying and mastering machine consciousness. We have shown that Turing Machine language and consciousness are independent. We've discovered boundaries separating the computability and non-computability of building different forms of *AMCDs* and have identified candidates for their characteristics that may be critical. In the general case of an *AMCD* as described in Definition 3.4, we have shown that it is necessary for this *AMCD* to itself be capable of experiencing consciousness, if it is possible to build one at all. If machine consciousness is possible and not ubiquitous as the panpsychists believe as per Assumption 3.1, then no matter what, nobody even in the distant future, regardless of how advanced their technology is, will ever be able to build an infallible *AMCD* that itself is not capable of experiencing consciousness. This leaves the door open for the possibility of creating a conscious *AMCD* M_C , but even if this is possible, how would we know that M_C is conscious? We would first need another conscious *AMCD* to determine that M_C is conscious. But then how would we know that one is conscious, and so on, thus creating an infinite regress and a prime mover problem.

In the case of N-Step *AMCDs* described in Definition 4.3, we have shown that the consciousness of the detector, the time that it is given to produce its output, as well as the ability to simulate are all boundary conditions that can be toggled to place the problem on one side of computability or the other. Unfortunately, our positive result showing that it is possible to build an N-Step *AMCD* is not immediately actionable because the assumptions underpinning it would require us to solve the 'Hard Problem of Consciousness' and also make considerable advancements in our computer simulation capabilities.

It is worth noting that the present results only apply to computational means of detecting consciousness. As described above in Section 1, science knows almost nothing about the nature of consciousness, including its underlying causal mechanisms. We don't know how to detect it, or even if it can be detected in an objective way - indeed, the present paper provides grounds for skepticism. This lack of knowledge on our part lies at the heart of the 'Hard Problem of Consciousness'. Does consciousness only arise as a consequence of the configuration of certain materials, as is the case with magnetism? Or is it substrate independent? It is conceivable that consciousness is a physical property associated with matter, like mass or magnetism, and it can therefore be detected by some physical means that we don't yet understand. Perhaps when present, consciousness creates a 'mental field' analogous to a magnetic or a gravitational field, or a form of radiation that we haven't yet discovered, in which case something akin to an (unconscious) Geiger counter for this type of radiation could be placed on someone's head or on a computer to objectively show that it experiences consciousness. The present results rule out certain computational possibilities for machine consciousness detection, but as far-fetched as these other physical possibilities may seem, we are not able to rule them out.

On the other hand, if there really are no computational nor physical methods to objectively detect consciousness, then unless there exists some third way, we may live in a Universe where general consciousness detection truly is impossible. In this case Turing was right all along and the best we can do is test for behavioral indistinguishability.

It is gratifying to prove results that unify two of Turing's great interests, namely the areas of com-

putability and machine consciousness. Why did Turing himself not combine the two areas back in the 1950s and instead chose to devise a more subjective version of the Turing Test? The proof of Theorem 4.3 above so closely parallels Turing’s own proof of the Halting Problem that one is tempted to conclude that without the hindsight benefit of fundamental papers such as [Put60, Arb61], his attention simply wasn’t focused in this direction, possibly for lack of a technical definition of consciousness, or possibly because the idea of conscious software would have been too exotic so many decades ago.

Computer science and AI have progressed considerably since then, but much more research is still needed to solve the biggest problems mentioned above and gain true insights into the nature of consciousness.

6 Related Open Problems, Conjectures, & Future Research

Rather than focusing on the major open problems in the area of consciousness studies such as its precise physical nature or whether conscious machines are even possible or what the difference between conscious and non-conscious computations are (assuming the panpsychists are wrong), we will instead highlight some more tactical research areas that are closely related to our present results:

6.1 Machine Consciousness & Languages

In Section 4.2 we showed that it is trivially easy to take any non-conscious machine and turn it into a conscious one that accepts exactly the same language, but the opposite direction of this statement remains a significant open problem. For any arbitrary conscious machine, can we ‘downgrade’ it to be a non-conscious ‘philosophical zombie’ version? In other words, is there an equivalent machine that decides exactly the same language but is not conscious? Closely related to this problem is one asking whether every artificial superintelligence must be conscious. We are unaware of a compelling intuition in either direction on these questions and therefore decline to hazard a conjecture.

6.2 Generalization of The AMCD Problem

However, it is tempting to conjecture that the obvious generalization of Theorem 4.3 is true. In this result, we were only able to show that one cannot detect machine consciousness with an *unconscious AMCD*, but in all likelihood it isn’t possible to create an *AMCD*, whether conscious or not, because the detection of almost any non-trivial property of a Turing Machine by another machine is typically not computable. As discussed above in Section 4.1, Rice’s Theorem technically does not apply here, but that does not mean that it cannot guide our intuitions. Together with Theorem 4.3, this suggests that the following conjecture is likely true, and proving it would constitute a valuable extension of the present research:

Conjecture 6.1. *If Assumption 3.1 holds true, then it is not possible to create an Automated Machine Consciousness Detector M_C , whether it is capable of experiencing consciousness or not.*

6.3 Generalization of The N-Step AMCD Problem

More research into the computability of N-Step *AMCDs* is also needed. In Theorem 4.5, we simultaneously made the problem more difficult in two ways, by both disallowing M_C to be conscious, and also by restricting the number of steps it is allowed to perform below n . The Scientific Method dictates that we should change only one variable at a time, so it would be preferable to prove this result with only one change or the other. Since both of these changes remove the ability to use simulation as a tool, it is tempting to conjecture that both resulting problems would remain incomputable.

If we allow M_C to be conscious, then is it possible for it to do its work in under n steps? That is the purpose of the following conjecture:

Conjecture 6.2. *Under the premise that Assumption 3.1 holds true, it is not possible to build an N -Step Automated Machine Consciousness Detector M_C (whether conscious or not) that computes its output within $n - \epsilon$ steps, where n is described in Definition 4.3, and ϵ is a constant.*

Conversely, is this problem solvable if we disallow M_C from being conscious, but do allow it the full n steps in which to do its work? Again we hypothesize that the answer is no:

Conjecture 6.3. *Under the premise that Assumption 3.1 holds true, it is not possible to build an N -Step Automated Machine Consciousness Detector M_C that is itself not capable of experiencing consciousness.*

6.4 Generalization of Proof Technique

In Sections 4.3 and 4.4.2, we respectively proved Theorems 4.3 and 4.5 and we generalized these incomputability results to include all qualia as well as more absurd examples of computations that summon an apparition of Napoleon Bonaparte. It is clear that this proof technique is very general like Rice’s Theorem, and that it might be possible to generalize it further. What is less clear is precisely what the largest class of properties to which we can extend these results might be. If the Computational Theory of Mind is correct, then certain computations seem to have the strange side effect of summoning or manifesting or conjuring a consciousness into existence, not unlike our example of Napoleon’s ghost. For this reason it perhaps isn’t surprising that consciousness as well as the human soul have been described as the ‘ghost in the machine’.

If this larger class exists and it is possible to precisely define it, this may help us to better understand the nature of consciousness itself.

6.5 Implications For The Physical Nature of Consciousness

Another research direction is suggested by our positive result in Theorem 4.4. In it, we connected the physical and theoretical worlds through computer simulation. Can this technique be used to ‘boost’ negative theoretical results into the real world and thereby allow us to draw conclusions about the physical nature of consciousness? More specifically, if we assume that the physical nature of consciousness is such that it can be detected by some type of physical sensor S , and if we can use a simulated form of S to build a consciousness detector in a simulation that contradicts a known incomputability result, then that contradiction in the simulated digital world would mean that the nature of consciousness in the physical world is such that it cannot be detected by any physical sensor. Any result along these lines ruling out such possibilities would constitute a *major* breakthrough in our understanding of consciousness and progress towards solving the Hard Problem.

Acknowledgments

TODO: THANK EVERYONE WHO PROVIDED COMMENTS

References

- [Aar13] S. Aaronson. The Ghost In The Quantum Turing Machine. URL: <https://www.scottaaronson.com/papers/giqtm3.pdf>, 2013.
- [ACA⁺21] M. Alfonseca, M. Cebrián, A.F. Anta, L. Coviello, A. Abeliuk, and I. Rahwan. Superintelligence Cannot Be Contained: Lessons From Computability Theory. *Journal of Artificial Intelligence Research*, 70:65 – 76, 2021.
- [Arb61] M. Arbib. Turing Machines, Finite Automata, and Neural Nets. *Journal of The ACM*, Vol. 8 Issue 2:467 – 475, 1961.

- [Baa88] B. Baars. *A Cognitive Theory of Consciousness*. Cambridge University Press, Cambridge, UK, 1988.
- [BB22] L. Blum and M. Blum. A Theory of Consciousness From A Theoretical Computer Science Perspective: Insights From The Conscious Turing Machine. *Proceedings of The National Academy of Sciences of The United States of America (PNAS)*, Vol. 119, No. 21, 2022.
- [Cas02] V. Caston. Aristotle on Consciousness. *Mind*, Vol. 111 No.444:751 – 815, 2002.
- [Cha96] D. Chalmers. *The Conscious Mind: In Search Of A Fundamental Theory*. Oxford University Press, New York, 1996.
- [Cha07] D. Chalmers. The Hard Problem of Consciousness. In M. Velmans and S. Schneider, editors, *The Blackwell Companion To Consciousness*, pages 225 – 235. Blackwell Publishing, 2007.
- [DSH21] A. Doerig, A. Schurger, and M. H. Herzog. Hard Criteria For Empirical Theories of Consciousness. *Cognitive Neuroscience*, Vol. 12 No. 2:41 – 62, 2021.
- [Fag18] D. Faggella. Could Artificial Intelligence Become Conscious? 33 Researchers Contribute Their Opinion. URL: <https://emerj.com/ai-market-research/conscious-artificial-intelligence/>, 2018. Available online at Emerj.
- [Gra22] N. Grant. Google Fires Engineer Who Claims Its A.I. Is Conscious. *New York Times*, July 2022.
- [Hai19] P. O. Haikonen. *Consciousness and Robot Sentience, Second Edition*. World Scientific Publishing Co., Singapore, 2019.
- [Hai20] P. O. Haikonen. On Artificial Intelligence and Consciousness. *Journal of Artificial Intelligence and Consciousness*, Vol. 7, No. 1:73 – 82, 2020.
- [Har91] S. Harnad. Other Bodies, Other Minds: A Machine Incarnation of An Old Philosophical Problem. *Minds and Machines*, 1:43 – 54, 1991.
- [Har19] A. Harris. *Conscious*. HarperCollins, New York, 2019.
- [Kle24] J. Kleiner. Towards A Structural Turn in Consciousness Science. *Consciousness and Cognition*, 119:103653, 2024.
- [Low12] P. Low. Cambridge Declaration on Consciousness. In *Proceedings of the Francis Crick Memorial Conference*, pages 1 – 2. Cambridge University, 2012.
- [Mas16] J. W. D. Mason. Quasi-conscious Multivariate Systems. *Complexity*, Vol. 21 No. S1:125 – 147, 2016.
- [Mas21] J. W. D. Mason. Model Unity and The Unity of Consciousness: Developments in Expected Float Entropy Minimisation. *Entropy*, Vol. 22 Issue 11:1444 – 1465, 2021.
- [McC79] J. McCarthy. Ascribing Mental Qualities to Machines. In M. Ringle, editor, *Philosophical Perspectives in Artificial Intelligence*, page 161. Humanities Press, Atlantic Highlands, N.J., 1979.
- [MP43] W. S. McCulloch and W. Pitts. A Logical Calculus of The Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*, Vol. 5:115 – 133, 1943.
- [Neu58] J. Von Neumann. *The Computer and The Brain*. Yale University Press, New Haven, CT, 1958.

- [Pen89] R. Penrose. *The Emperor's New Mind: Concerning Computers, Minds, and the Laws of Physics*. Oxford University Press, 1989.
- [Pen94] R. Penrose. *Shadows of the Mind: A Search for the Missing Science of Consciousness*. Oxford University Press, 1994.
- [Put60] H. Putnam. Minds and Machines. In Sidney Hook, editor, *Dimensions Of Mind: A Symposium*, pages 138 – 164. New York University Press, 1960.
- [Reg14] J. A. Reggia. Conscious Machines: The AI Perspective. *The Nature of Humans and Machines - A Multidisciplinary Discourse: Papers From The 2014 AAAI Fall Symposium*, 2014.
- [Ric53] H. G. Rice. Classes of Recursively Enumerable Sets and Their Decision Problems. *Transactions of The American Mathematical Society*, Vol. 74, No. 2:358 – 366, 1953.
- [Sea80] J. Searle. Minds, Brains, and Programs. *Behavioral and Brain Sciences*, 3(3):417 – 457, 1980.
- [Sip97] M. Sipser. *Introduction to The Theory of Computation, First Edition*. PWS Publishing Company, Boston, MA, 1997.
- [Sip13] M. Sipser. *Introduction to The Theory of Computation, Third Edition*. Cengage Learning, Boston, MA, 2013.
- [SMB⁺21] D. Sattin, F. G. Magnani, L. Bartesaghi, Milena Caputo, A. V. Fittipaldo, m. Cacciato, M. Picozzi, and M. Leonardi. Theoretical Models of Consciousness: A Scoping Review. *Brain Sciences*, 11:1 – 58, 2021.
- [SSP21] C. M. Signorelli, J. Szczotka, and R. Prentner. Explanatory Profiles of Models of Consciousness - Towards A Systematic Classification. *Neuroscience of Consciousness*, Vol. 7 No. 2:1 – 13, 2021.
- [Tay99] C. C. W. Taylor. *The Atomists Leucippus and Democritus: Fragments, A Text and Translation with Commentary*. University of Toronto Press, Toronto, 1999.
- [TK15] G. Tononi and C. Koch. Consciousness: Here, There, and Everywhere? *Philosophical Transactions of The Royal Society B*, Vol. 370, No. 1668:20140167, 2015.
- [Tur37] A. M. Turing. On Computable Numbers, With An Application To The Entscheidungsproblem. *Proceedings of The London Mathematical Society, Series 2*, Vol. 42:230 – 265, 1937.
- [Tur50] A. M. Turing. Computing Machinery and Intelligence. *Mind, New Series*, Vol. 59, No. 236:433 – 460, 1950.
- [Tye21] Michael Tye. Qualia. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Fall 2021 edition, 2021.
- [Wik24] Wikipedia. Eugene Goostman - Wikipedia, The Free Encyclopedia. URL: https://en.wikipedia.org/wiki/Eugene_Goostman, 2024. [Online; Accessed 23-March-2024].