Will AI Succeed? The "Yes" Position

William J. Rapaport

**Department of Computer Science and Engineering,
Department of Philosophy, Department of Linguistics,
and Center for Cognitive Science
University at Buffalo, The State University of New York,
Buffalo, NY 14260-2500**

rapaport@buffalo.edu
http://www.cse.buffalo.edu/~rapaport/

December 5, 2024

Abstract

This is a draft of the "Yes" side of a proposed debate book, *Will AI Succeed?*. The "No" position will be taken by Selmer Bringsjord, and will be followed by rejoinders on each side.

AI should be considered as the branch of computer science that investigates whether, and to what extent, cognition is computable. Computability is a logical or mathematical notion. So, the only way to prove that something—including (some aspect of) cognition—is *not* computable is via a logical or mathematical argument. Because no such argument has met with general acceptance (in the way that other proofs of non-computability—such as the Halting Problem—have been generally accepted), there is no logical reason to think that AI *won't* eventually match human intelligence. Along the way, I discuss the Turing Test as a measure of AI's success at showing the computability of various aspects of cognition, and I consider the potential roadblocks set by consciousness, qualia, and mathematical intuition.

Contents

1 Yes, (Real) Intelligence (Probably) Is (Artificially) Computable 5

2 What Is AI? 6
2.1 A Very, Very Brief History of AI . . . 6
2.1.1 Turing on "Machine" Intelligence . . . 6
2.1.2 The Dartmouth Project . . . 7
2.2 Other Definitions of 'AI' . . . 8
2.2.1 Minsky's Definition . . . 8
2.2.2 Boden's Definition . . . 8
2.3 'Artificial' and 'Intelligence' . . . 9
2.3.1 Intelligence? Or Cognition! . . . 9
2.3.2 Artificial? Synthetic? Or Computational! . . . 12
2.3.3 Does 'Artificial' *Have* to Mean "Computable"? . . . 13
2.4 AI as Computational Cognition . . . 15
2.4.1 AI and Computer Science . . . 15
2.4.2 Algorithms . . . 17
2.4.3 Computability . . . 18
2.4.4 Cognition . . . 18
2.5 AI as the Popular Press Sees It . . . 19
2.6 Three Goals of AI . . . 23
2.7 Summary . . . 24

3 What Counts as "Success"? 24
3.1 Theoretical vs. Practical Success . . . 25
3.2 Measuring and Evaluating Success . . . 26
3.2.1 Measuring Devices . . . 26
3.2.2 The Turing Test . . . 27
3.2.3 AI's Successes . . . 32
3.3 Summary . . . 33

4 What Counts as 'Failure'? 34
4.1 Practical Arguments against AI's Success . . . 34
4.2 Biological Arguments against AI's Success . . . 36
4.2.1 Hubert Dreyfus's Argument . . . 36
4.2.2 John Searle's Argument . . . 37
4.3 Landgrebe and Smith's Argument . . . 39
4.4 Vitalistic Arguments against AI's Success . . . 40
4.5 Mathematical Arguments against AI's Success . . . 44
4.6 Summary . . . 47

5 Will *All* of AI Succeed? 47
5.1 The Cognitive Computability Thesis . . . 47
5.2 Is Consciousness Computable? . . . 49
5.3 Are Qualia Computable? . . . 51
5.4 Is Mathematical Cognition Computable? . . . 54
5.4.1 Gödel's Incompleteness Theorem . . . 54
5.4.2 Lucas's Argument . . . 56
5.4.3 Turing's Argument . . . 58

5.5 Summary: But *X* Is Not Computable! . . . 60

6 Concluding Remarks 61

7 Final Summary 64

A Sidebar: The Chinese Room Argument 65

B Sidebar: Computability 67
B.1 Varieties of Computability . . . 67
B.2 Turing Machine Computability . . . 69
B.2.1 Five Great Insights of Computer Science . . . 69
B.2.2 Turing Machines . . . 70
B.2.3 Universal Turing Machines . . . 71
B.3 Efficient Computability . . . 71

C Sidebar: Simulations 73

D Sidebar: Recursion 76

E Sidebar: *Reductio* Arguments 78

F Sidebar: The Halting Problem 80
F.1 The Halting Function *H* . . . 80
F.2 Proof Sketch that *H* Is Not Computable . . . 82
F.2.1 Step 1 . . . 82
F.2.2 Step 2 . . . 83
F.2.3 Step 3 . . . 84
F.2.4 Step 4 . . . 84
F.2.5 Final Result . . . 85

G Glossary 86

Computers can be programmed, and have been programmed, to simulate at a symbolic level the processes that are used in human thinking. We need not talk about computers thinking in the future tense; they have b een thinking (in smaller or bigger ways) for 35 years [i.e., since about 1960]. They have been thinking "logically" and they have been thinking "intuitively"—even "creatively."
—Herbert S. Simon (1995, p. 38)

Already, artificial intelligence is integrally involved in solving problems both in science and in daily human life: designing molecules to fight infectious diseases; discovering long-lost archaeological sites; identifying people at high risk of committing suicide.

These systems are a marvel and can make our lives easier. But ultimately they are as flawed as we are, because we are behind all of it. We choose what data to train A.I. on, unwittingly passing along our unspoken biases about one another. And we are all too quick to think that our machines, being machines, are smarter and more objective than us.

Is this humility or hubris? Do we underrate human intelligence, or do we take too much pride in it, assuming that in our cleverness ... we can create things more clever still? Perhaps a cleareyed view is best. "*Remove the romanticism*," the computer engineer Jaron Lanier tells Dennis Overbye, our cosmic correspondent. "*It's not a creature like a cat, it's just an algorithm running*."
—Alan Burdick (2020, my italics)

[S. Ovide:] Well, are these things [AI computers] going to get as capable or more capable than people?

[C. Metz:] There is a massive argument about this, even among artificial intelligence researchers. Some of them believe that if you give neural networks enough time and data or computerized simulations of the world, eventually they'll reach human intelligence. And others think that's just ridiculous, at least in the foreseeable future.
—Cade Metz, as interviewed by Shira Ovide (2020).

1 Yes, (Real) Intelligence (Probably) Is (Artificially) Computable

> … there is no obstacle *in principle* to human-level AI intelligence …
> —Margaret Boden (2018, p. 136)

The idea of building intelligent robots has been around for hundreds, if not thousands, of years. Jewish mythology from at least the 16th century tells of the legend of the Golem, a statue that comes to life when certain Hebrew words are inscribed on it. As Ted Chiang's 2002 story "Seventy-Two Letters" suggests, those words can be thought of as the computer program for a robot. The 17th-century French philosopher René Descartes (1637) discussed the idea that non-human animals were "automata": automatic machines that behaved as if they had minds. And the Czech playwright Karel Čapek's 1920 play *R.U.R.* introduced the term 'robot'. Artificial Intelligence (AI) is the modern, scientific descendent of these.

Will AI match (or possibly exceed) human intelligence? Put more succinctly, will AI succeed? The answer to this question depends, of course, on what we mean by 'AI' and what we mean by 'succeed'.[1] So we first have to answer two other questions: What is AI? And what would it mean for AI to be successful? (It may even depend on what we mean by 'will'! On that question, see §§3.1 and 4.1.)

After telling you how I view AI and what I think success might look like, I will argue for a scientifically optimistic view, not exactly that AI *will* succeed, but that

> **There is currently no known logical or mathematical reason to think that AI *won't* succeed.**

(I will not be arguing about *physical* limitations on AI's success, but see my comments in §3.1.) To paraphrase what the philosopher Georges Rey (1986, p. 179) said about the anti-AI "Chinese Room Argument" (see Sidebar A),

> I don't mean to suggest that it would be easy to construct particularly the … [AI] programs that … would [be] need[ed]; but I have yet to see an argument that shows that it would be *impossible*.

My argument will proceed as follows: I will first discuss the nature of AI, suggesting that it should be viewed as the branch of computer science that is concerned with computational theories of cognition. Next, we'll consider ways to measure the success of AI theories, primarily via Turing-like Tests. I'll then reply to various, standard arguments that AI won't or can't be accomplished. I'll try to

[1]I use *single* quotes to form names of words, and I use *double* quotes for meanings (of singly quoted words), for quotations, and as "scare" quotes.

show that none of them succeed, and that no one has successfully offered the only kind of argument—a logical one—that *could* succeed. Finally, we'll consider three features of cognition that *might* prove to be stumbling blocks for AI.

Sidebars and a glossary provide background on related issues for those unfamiliar with some of them, and . . .

. . . Further Reading boxes like this provide guidance to some of the literature on relevant topics:
On the Golem, see https://www.jewishvirtuallibrary.org/the-golem. The history of robots in literature from *The Iliad* on is surveyed by Daniel Mendelsohn in a 2015 review of the films *Her* and *Ex Machina*. See also https://en.wikipedia.org/wiki/Robot#Early_beginnings.

2 What Is AI?

[Artificial intelligence is] a term for a collection of concepts that allow computer systems to vaguely work like the brain.
—Cade Metz, as quoted in Ovide 2021

2.1 A Very, Very Brief History of AI

2.1.1 Turing on "Machine" Intelligence

AI and computer science share their history.

If the field of computer science can trace its origins back to mathematician Alan Turing's 1936 paper on computability (see Sidebar B), then AI should trace its origins back at least to some of Turing's speculations in the early 1940s on what he called "machine intelligence" (Copeland and Bowen, 2017, p. 9). The first publications on the topic were two papers by Turing, one on his ACE ("automatic computing engine") computer and one titled "Intelligent Machinery" (Turing, 1947, 1948). These publications were followed by his second most famous paper, "Computing Machinery and Intelligence" (Turing, 1950). The following year, he published another paper on the topic (Turing, 1951b), followed by two radio broadcasts on the ability of computers to "think" (Turing, 1951a; Turing et al., 1952). The closest he came to defining the field of machine intelligence might be this passage from Turing 1951a:

> . . . our main problem, how to programme a machine to *imitate* a brain, or as we might say more briefly, if less accurately, to think. (p. 485, my italics)

Arguably, Turing was not only the founder of computer science but also of AI.[2]

2.1.2 The Dartmouth Project

> The original sin of the A.I. pioneers was that they called it artificial intelligence.
> —Cade Metz, as quoted in Ovide 2021

AI—under that name—has a very precise starting point and initial definition. In 1955, two mathematicians (John McCarthy and Marvin L. Minsky) and two electrical engineers (Nathaniel Rochester and Claude E. Shannon) wrote a proposal for a "Summer Research Project on Artificial Intelligence" that was held the next year at Dartmouth College.

They characterized AI in two slightly different ways. At the beginning of the proposal, they wrote:

> The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a *machine* can be made to *simulate* it. (McCarthy et al., 1955, p. 1, my italics)

The "machine" was a digital computer, as it was for Turing. (But see §2.3.3 for other possibilities.)

At the end of the proposal, they qualified this:

> For the present purpose the artificial intelligence problem is taken to be that of making a machine behave in ways that would be *called* intelligent if a *human* were so behaving. (McCarthy et al., 1955, p. 12, my italics)

So, historically, the first definition of 'AI' was, roughly, the study of how to make (digital) computers simulate all aspects of intelligence—or behave in "intelligent" ways—where 'intelligence' is understood in terms of *human* behavior.

This raises an interesting question: If AI "succeeds", will the kind of "intelligence" it produces be "real" intelligence? Or will it be "just a simulation" (or an "imitation", to use Turing's term)? Not all things that computers do are "mere simulations": A computer that has been programmed to write music in the style of Bach (e.g., Ebcioğlu 1990; Hadjeres et al. 2017) doesn't just "simulate" writing music; it really does write music. Whether the music is really in the style of Bach and whether it is any good are entirely different matters! (For more on this, see Sidebar C: Simulations.)

[2]In addition, his 1952 paper, "The Chemical Basis of Morphogenesis", "laid the groundwork for the entire field of mathematical biology", according to geneticist Gregory S. Barsh, as quoted in Gorman (2021).

2.2 Other Definitions of 'AI'

Those initial definitions were "forward looking" in the sense that they set a goal of what AI would (or should) be. As the field progressed, other definitions were crafted in a "backward looking" fashion, based on what AI had actually become (Rapaport, 2020b). Two of these are worth considering.

2.2.1 Minsky's Definition

The first is by Minsky, who, as we just saw, was one of the pioneers of AI research:

> ... *artificial intelligence*, the science of making machines **do** things that would **require** intelligence if done by men [i.e., by humans]. (Minsky, 1968, p. v, my boldface)

Minsky's definition—as with McCarthy et al.'s second one—suggests that the methodology of AI is to study *human* intelligence in order to learn how to program *computers* to do intelligent things. The methodology of studying *humans* in order to learn what *computers* might be able to do was essentially Turing's methodology in his 1936 paper that initiated the study of computation: Turing investigated how humans performed mathematical computations, and then showed how a machine could do it in the same way (see Sidebar B: Computability). (For a step-by-step discussion of Turing's methodology, see Rapaport 2023a, Ch. 8.)

There is a subtle difference between the McCarthy et al. definition and Minsky's: Although neither tries to characterize what intelligence *is*, the former only talks about behaviors that we would *call* 'intelligent', whereas the latter talks about things that *require* intelligence. Note also that Minsky talks about (really) *doing* intelligent things, not merely *simulating* them.

2.2.2 Boden's Definition

The second "backward-looking" definition of 'AI' is by Margaret Boden, one of the pioneers of cognitive science:

> By "artificial intelligence" I ... mean the use of computer programs and programming techniques to cast light on the principles of intelligence in general and human thought in particular. (Boden, 1977, p. 5)

Boden's definition suggests a methodology that goes in the opposite direction of Minsky's: to study *computers* in order to learn something about *humans*. Her definition focuses more on *understanding* the nature of intelligence than on *simulating* or reproducing it. Turing advocated this approach, as well: "the attempt to make a thinking machine will help us greatly in finding out how we think ourselves" (Turing, 1951a, p. 486).

Philosopher Mark Sprevak (2017, p. 278) suggested that a definition like Minsky's is what *AI* is concerned with, whereas a definition like Boden's is more the province of *cognitive science* (see §2.4). That's an interesting distinction that I won't dispute. But I do believe that AI is, in fact, a two-way street: Both Minsky's (and Turing's) view of AI as moving from humans to computers as well as Boden's (and Turing's) view of it as moving from computers to humans are components of AI research (and, for that matter, of cognitive science research).

Further Reading:
Interestingly, Cassenti et al. (2022, §5) argue that "AI is best considered an engineering discipline" whose scientific basis is cognitive science. (Compare the discussion of computer science as an engineering discipline whose scientific basis is mathematics in Rapaport 2023a, §§3.15.2, 5.7.)

For a compendium of other definitions of AI, see https://cse.buffalo.edu/~rapaport/definitions.of.ai.html

2.3 'Artificial' and 'Intelligence'

Two further questions are raised by each of these definitions: What is meant by 'intelligence'? And why is the kind that AI aims at said to be "artificial"?

2.3.1 Intelligence? Or Cognition!

Let's begin with intelligence. One common use of 'intelligence' is in the phrase 'intelligence quotient' (IQ). Here, the suggestion is that intelligence ("high IQ") is something that only very bright people have. But that's not what's involved in AI.

What *is* involved? Facility at games like chess has often been suggested as a mark of intelligence: The great German writer Goethe said that chess was a "touchstone of human intelligence" (Copeland and Prinz, 2017, p. 329), and Turing thought that chess was a computable aspect of intelligence (Proudfoot, 2017a, p. 288).

Or consider the kinds of things that McCarthy et al. had in mind in 1955:

> An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. (p. 1)

An early AI program—exhibited at the 1956 conference—concerned logical reasoning. This program—the Logic Theorist (Newell et al., 1958)—proved theorems in propositional logic and was written by observing how humans proved such theorems, following the human-to-computer methodology. The anthology in which Minsky's definition appeared included AI studies of language (both grammar and meaning), memory, analogy, and common sense. Other topics that were early goals of AI researchers included visual perception, appearance and illusion, image recognition, learning, and understanding (Minsky and Papert, 1974).

All of these, along with such things as belief, consciousness, emotion, planning, problem solving, mental imagery, concepts, sensation, thought, etc., are better summed up by the term '*cognition*' than by 'intelligence'.

Sometimes other terms are used:

1. The traditional question is whether computers can "*think*", but thinking is just one kind of cognition.

2. The philosopher John Searle talks about computers having "*intentionality*". That's a technical philosophical term with a variety of meanings. Sometimes, minds are said to be intentional in that thoughts are always *directed to an object*: When you think about someone, that person is the object of your thought; when you believe that $2 + 2 = 4$, that mathematical proposition is the object of your thought. Note that you can also think about things that *don't* exist, and you can believe things that are *false*, such as that Santa Claus is skinny. But intentionality in this technical sense refers to the facts that, whenever we think, we always think *about something*, and that, whenever we believe, we always believe *some proposition*. These are all aspects of cognition, and we would certainly want AI programs to be intentional in this sense.

3. A related sense of 'intentionality' is as a synonym for mentality or '*understanding*'. This, too, is an aspect of cognition, though whether an AI computer would (or could) "really" understand is related to the simulation issue. (See §4.2.2, Sidebar A: The Chinese Room Argument, and Sidebar C: Simulation.)

4. Yet another sense of 'intentionality' can be found in the philosopher Daniel C. Dennett's (1971) notion of the "Intentional stance"—a way of understanding a systems' behavior in terms of its beliefs, desires, and goals. (See §4.4.)

5. Finally, another term that is often used is '*consciousness*', yet another aspect of cognition, which we'll discuss in §5.2.

As can be seen from some of the definitions of AI, the kind of cognition that is commonly taken as the model is *human* cognition. But the study of *non-human* animal cognition is also a legitimate part of AI. Even if you don't think that non-human animals are "intelligent" (or as intelligent as humans), surely they exhibit cognitive behaviors. Suppose that dinosaurs had not gone extinct and that *Homo sapiens* had not evolved. Would "intelligent" life have evolved? Such a question seems to assume that 'intelligence' refers to *human* (or human-like) cognition, including language, extensive and creative tool use, extensive artifact construction, etc. But dinosaur descendents might have evolved a kind of intelligence with little or none of these features. So intelligence need not be identified with *human* intelligence. There could be a *range* of cognitive features (including others unimagined by us). Surely, elephants, for example, are "intelligent" in ways that we humans are not, yet their intelligence should fit in that range.

Moreover, any theory of cognition (artificial, computational, or whatever) should also allow for the possibility of *non-animal* (e.g., extra-terrestrial or robotic) cognition:

> Imagine that a party of extra-terrestrials find their way to Earth and impress us with their mathematics and poetry. We discover they have no organ resembling a human brain: inside they are just a seething mixture of gases. Does the fact that these hypothetical aliens contain nothing like human brain cells imply that they do not think? Or is their mathematics and poetry proof enough that they *must* think—and so also proof that the mammalian brain is not the only way of doing whatever it is that we call thinking? (Copeland, 2017b, p. 268)

These are other reasons to prefer the term 'cognition' over 'intelligence'.

In any case, the question of whether AI will "succeed" is usually taken to be about whether AI will meet or exceed *human*-level cognition.

Further Reading:
An excellent survey of definitions of 'intelligence' in general and of AI in particular is Legg and Hutter 2007. A recent discussion of the lack of a characterization of intelligence can be found in P. Wang 2019, with commentary in Monett et al. 2020. Other useful general discussions of intelligence include Gardner 1983; Sternberg 1985; Smith 2019; Dietrich et al. 2021.

On AI and IQ, see my "Artificial I.Q. Test" at http://www.cse.buffalo.edu/~rapaport/AIQ/aiq.html (Rapaport, 1986b), as well as Ohlsson et al. 2015.

On common sense, see Shanahan et al. 2020.

My characterization of intentionality in this section is not intended to be historically accurate with respect to the philosopher Franz Brentano—who took it as the "mark of the mental"—but only to the usual contemporary interpretation of intentionality. For the history, see Brentano 1874; Kriegel 2016; Jacob 2019. For the distinction between 'intentionality' spelled with a 't' and another technical philosophical term, 'intensionality' spelled with an 's', see https://cse.buffalo.edu/~rapaport/intensional.html. For Searle's comments that suggest the "understanding" interpretation of 'intentionality', see Searle 1980, §II, p. 420; p. 424, fn. 2. Boden 2018, p. 119, is more explicit about this interpretation.

On non-*mammalian* animal brains that can think, see Tye 2017. On animal intelligence in general and the range of intelligence, see Darwiche 2018; Crosby 2020; Hernández-Orallo 2020; Shanahan et al. 2020; Halina 2021. Griffiths 2020 discusses differences between extra-terrestrial, human, and artificial intelligence. Donald Davidson 1990, p. 3, made a similar observation to that of Copeland: "If I were to discover that my best friend had been born by hatching from an egg, or had been conceived by a process that required the active collaboration of three creatures of different sexes, it would probably not influence my opinion that he or she could think." (His "collaboration" example may have been inspired by Asimov 1972.) For more on what can be called "the space of possible minds", see Sloman 1984. See also the discussion of "how animals perceive our shared world" in Kolbert 2022.

On the importance of human-level cognition in AI research, see, e.g., Brown et al. 2020, esp. p. 7, in which the GPT-3 computational linguistic model is measured against *human* performance. For general remarks on human vs. animal vs. AI intelligence, especially in connection with "deep machine learning" (see §2.5, below), see Buckner 2013, 2020.

2.3.2 Artificial? Synthetic? Or Computational!

What about artificiality? The adjective 'artificial' often suggests that the thing that it characterizes is not "the real thing": An artificial flower might be made of paper or plastic; it only looks like a flower but would typically not feel or smell like a real flower, and is certainly not a living plant. But this is not the kind of artificiality that AI is concerned with.

A better adjective would be 'synthetic': A synthetic diamond *is* a real diamond in terms of its physical and chemical properties, except that it is created in a laboratory or factory, rather than formed by "natural" processes and found in the earth (Sullivan, 2018). So, it would be more accurate to say that AI is concerned with *synthetic* intelligence: "real" intelligence that is created in a lab.

But, in the case of AI, an even better adjective is 'computational', because the kind of intelligence that AI is concerned to synthesize (i.e., duplicate or, if you prefer, simulate) is one that can be described in computational terms so that a computer could be programmed to exhibit it. I'll make this more precise in §2.4.

2.3.3 Does 'Artificial' *Have* to Mean "Computable"?

I am taking the 'A' of 'AI' to refer to computability. But there is a more general sense of "artificiality" that is neither restricted to computability nor is a synonym of 'fake'. To be artificial in this more general sense is, roughly, to be manufactured: An "artifact" is usually considered to be something made by humans, such as furniture, houses, cars, etc., rather than found in nature. (But aren't bird's nests and beehives also similarly "manufactured", albeit not by humans? Yet they are considered to be part of "nature".)

So AI could be understood as the *mechanical* simulation or reproduction of intelligence or cognition, not necessarily solely by digital computational means. For instance, it might include a different kind of computation, such as analog, quantum, or DNA computation. There are also "hybrid" systems consisting of humans working in tandem with computers. Some of these *might* provide more powerful (or at least different) methods than digital computation. Arguably, such hybrid systems are even *more* likely to succeed than AI understood solely in terms of computation. So, even if AI as (digital) computational cognition fails, AI more broadly construed might succeed. But this is a different issue than the one I am addressing here. (We'll return to this at the end of §2.5.)

There are problems, however, with extending the notion of "artificiality" beyond just digital computation. Suppose that AI is understood as the study of using computational methods *plus something else* to achieve "intelligence"—such as consideration of the chemistry or other physical aspects of its implementation. Such an understanding still treats the results as being "artificial" rather than as "natural" in the broader sense that I suggested. One problem is that such extensions beyond Turing computation might asymptotically approach being considered as *natural*, rather than artificial, cognition. Another is that they become too far removed from being purely mechanical, or they remain computational but are "opaque" and so yield no *understanding* of cognition (see §2.5). After all, there is a well-established, purely *natural* way to achieve the creation of an intelligent agent, which has been available since *Homo sapiens* evolved! By itself, that natural, biological means of creating a cognitive agent also yields no understanding of cognition, which is part of the reason why AI has proved so difficult. (We'll return to this point in §6.)

Further Reading:
On the nature of artifacts, see Dipert 1993; Mizoguchi and Kitamura 2009; Hilpinen 2011; and the cartoon at http://abstrusegoose.com/215. On artifacts in computer science, see Irmak 2012; Turner 2018.

On analog computation, see Rubinoff 1953; Samuel 1953, p. 1224, § "The Analogue Machine"; Jackson 1960; Montague 1960; Pour-El 1974; Moor 1978; Haugeland 1981; Copeland 1997, "Nonclassical Analog Computing Machines", pp. 699–704; Shagrir 1997, §§4.2, 4.3; Shagrir 1999; Holst 2000; Piccinini 2004, 2007, 2008, 2009, 2010, 2011, 2012; Care 2007; Zenil and Hernández-Quiroz 2007, especially p. 5; Fortnow 2010; Piccinini and Craver 2011; McMillan 2013; Corry 2017; Lee et al. 2022.

On quantum computing, see Hayes 1995, 2014; Grover 1999; Aaronson 2008, 2011, 2014; Monroe and Wineland 2008; Bacon 2010; Bacon and van Dam 2010; Piccinini and Anderson 2020.

On DNA and other forms of biological computing, see Adleman 1998; Shapiro and Benenson 2006; Qian and Winfree 2011; Lu and Purcell 2016.

Computer scientist Ben Shneiderman is an advocate of hybrid computer-human systems; see the inteview with him in Markoff 2020. And "partially" automated vehicles are a good example of them; see Casner et al. 2016 and the discussion in Rapaport 2023a, Ch. 17. But see Shladover 2016 for an argument that hybrid human-computer automated vehicles might be *harder* to achieve than fully automated ones. For more on such computer-human hybrid systems and the associated notion of "distributed cognition", see Hutchins 1995a,b, 2010; Hollan et al. 2000.

2.4 AI as Computational Cognition

For these reasons, I prefer the phrase ‘computational cognition’ to ‘artificial intelligence’ (though I will continue to refer to it as ‘AI’). This is on the model of such disciplines as computational linguistics, computational biology, computational chemistry, etc. In general, “computational *X*” uses computational methods to study *X*.

2.4.1 AI and Computer Science

I see AI as a branch of computer science, so what is computer science? In brief, I take computer science to be the scientific study of which problems can be solved, which tasks can be accomplished, and which features of the world can be understood computationally, and then to provide algorithms to show how this can be done efficiently, practically, physically, and ethically.

‘Scientific’ is to be understood as including science, technology, engineering, and mathematics (the “STEM” disciplines), so as not to beg any questions about whether computer science is really a science, a branch of engineering, or something else. By ‘efficiently’, I have in mind the considerations studied in the field of computational complexity (Aaronson, 2013), roughly, how much time and storage space does a computation require? In addition to complexity issues, “practicality” also includes such considerations as real-time computing and heuristic computing (see §3.1). “Physical” issues include the engineering of real computers. And “ethical” issues focus on what *should* be computed, as opposed to what *could* be computed, as well as focusing in such things as eliminating bias from computer programs (see §2.5).

Even more simply and elegantly,

> The Holy Grail of computer science is to capture the messy complexity of the natural world and express it algorithmically. (Teresa Marrin Nakra, quoted in J. Davidson 2006, p. 66)

Herbert Simon (1996b, p. 1) said something similar: “The central task of a natural science is … to show that complexity, correctly viewed, is only a mask for simplicity; to find pattern hidden in apparent chaos.”

So, I will understand AI as the branch of computer science that tries to capture the messy complexity of *cognition* and express it algorithmically:

AI is the computational study of cognition.

In short, it asks: Is cognition computable? It should also be noted that, if cognition is computable (or, more cautiously, to the extent that some aspects of cognition are computable), then anything that implements such cognitive algorithms will *be* cognitive: A computer that implements arithmetic algorithms really does arithmetic; a computer that implements visual-perception algorithms really sees; a computer that implements algorithms for logical reasoning really does logic (Rapaport, 1998, 2000b). (See also Sidebar C.)

Further Reading:
For an elaboration and general discussion of this characterization of computer science, see Shapiro 1995; Rapaport 2017c; Rapaport 2023a, Ch. 20. On the notion of what "should" be computed, see Arden 1980, p. 29; Tedre 2015, pp. 167–168. For more on ethical considerations in AI, see the first paragraph of Dennett 1985; Kim 2015; Bringsjord 2021; Dietrich et al. 2021, Ch. 8; and Awad et al. 2022. For a discussion that is applicable to AI of the importance of ethical considerations in scientific (specifically genetic) research, see de Souza 2021. Johnson and Verdicchio 2023 argue that ethical AI might not be computable.

Proudfoot 2017b, p. 305, argues that Turing would not have agreed with this characterization of AI, on the grounds that he viewed intelligence as a property that is *ascribed to* someone (or something) by an observer (e.g., an interrogator in a Turing Test), not as an intrinsic property of that person or thing. Perhaps; yet he was certainly interested in developing computers that do intelligent things like play chess, learn, etc. Moreover, an AI would be "intelligent" even if there were no observers to call them 'intelligent'. For discussion of this notion of "response-dependent" properties, see Moor 1976, p. 251; D. Davidson 1990, p. 4; Pettit 1991; Jackson and Pettit 2002; Proudfoot 2013; Wheeler 2020; Kryven et al. 2021. See also the Further Reading box at the end of §3.2.2. For related notions, see McCarthy 1979 and Dennett's notion of the "intentional stance" (Dennett 1987, 2009b and Dennett 2013a, Ch. 18), as well as the discussion in Rapaport 2023a, §12.4.1.

Pei Wang 2020 disagrees with my definition of 'AI'; see Sidebar B, fn. 30.

2.4.2 Algorithms

But what does it mean to say that something is "computable"? And what is an "algorithm"? Informally,

- an *algorithm* A for executor E to accomplish goal G is
  1. a procedure that
  2. takes a finite amount of time (i.e., A halts) and that
  3. ends with G accomplished, where:
- a *procedure* is a finite set (or sequence) of "statements" (or "rules", or "instructions"), such that:
- each statement S
  1. is composed of a finite number of symbols (or uninterpreted marks) from a finite alphabet and
  2. is "unambiguous" for E, where:
- S is unambiguous for E means that:
  1. E "knows how" to do S
     (i.e., execute the statement, follow the rule, carry out the instruction),
  2. E can do S,
  3. S can be done in a finite amount of time,
  4. and, after doing S, E "knows" what to do next

Algorithms are abstract mathematical entities that—typically—can be expressed by (i.e., implemented in) computer programs.

Further Reading:
That algorithms halt probably follows from procedures being a finite set and each statement being done in a finite amount of time. On infinite loops, see §4.5 and Sidebar F: The Halting Problem.

For a more detailed and general discussion of what an algorithm is, see any textbook on the theory of computation (e.g., Clark and Cowell 1976; Homer and Selman 2011; Soare 2016) or Rapaport 2023a, Ch. 7. For an exploration of whether a goal is really needed, see Rapaport 2017a or Rapaport 2023a, Ch. 16.

On implementation, see Pylyshyn 1984, pp. 74–78; Scheutz 1998; Sloman 1998; Rapaport 1999, 2005; Shagrir 2012; Rescorla 2013, 2014; Sprevak 2018.

2.4.3 Computability

Computability has to do with algorithms for mathematical functions. A mathematical function "from" a set of inputs "to" a set of outputs is a set of input-output *pairs* of members of the two sets such that no input corresponds to more than one output (i.e., the same input always yields the same output). To say that such a function is *computable* is to say that there is an algorithm that takes an input to that function and (typically) *manipulates* that input to calculate (i.e., to compute) its output. One standard *mathematical* model of computation is that of the Turing Machine. Others include Kurt Gödel's (and others') theory of general recursive functions and Alonzo Church's theory of the lambda calculus, all of which are mutually logically equivalent. A standard *physical* example is your laptop computer (or even your smartphone). (For more details, see Sidebar B: Computability.)

Further Reading:
I discuss the history and the mathematical details of computability in Rapaport 2023a, Chs. 7–8. For other notions of computability, see Shagrir 1997; Piccinini 2015, 2020. On physical computation, see Piccinini 2015; Duwell 2021; Shagrir 2022.

2.4.4 Cognition

Both AI and cognitive science began when psychologists and others rejected the "behaviorist" methodology that was then in vogue: studying human cognitive behavior solely in terms of stimuli and responses. In psychology, behaviorism focused only on such inputs and outputs: Pavlov's famous experiment input the stimulus of the ringing of a bell to a dog, and the dog output a response of saliva. But Pavlov didn't ask how the input and output were connected. (He didn't ask how the dog "computed" the bell-saliva "function".) Finding the *processes* that *intervene* between a stimulus and a response requires algorithmic thinking, and results in algorithms that specify the transformational relations between input and output. *Cognitive* psychology focused on those intervening algorithms. (None of this is to suggest that the bell-saliva "function" and "algorithm" are *cognitive* ones!)

So, an aspect of cognition will be computable if we can construct a computer program whose input-output behavior matches a human's cognitive input-output behavior, i.e., that will behave the way a human (or other cognitive entity) who exhibits that aspect of cognition behaves. The working assumption of AI is that *all aspects of cognition are computable*. So, in one sense, AI will "succeed" to the extent that it can show that all aspects of cognition are computable. Is the electrochemistry of neurons an aspect of cognition? On the present understanding of cognition as purely psychological, not physical, the answer is: no; neural electrochemistry is an aspect of the *implementation* of cognition.[3] (We'll go into more detail on this in subsequent sections.)

There are two more preliminary issues to get out of the way: the idea of AI in the popular press, and the different goals of AI researchers.

Further Reading:
For one of the earliest and clearest presentations of the algorithmic approach to cognition, see Miller et al. 1960. A precursor, explicitly based on Turing Machines, was McCulloch and Pitts 1943; for discussion, see Piccinini 2004.

2.5 AI as the Popular Press Sees It

> ... natural language processing [is] a branch of *machine learning* that uses algorithms to analyze the problems of human expression ...
> —Brooke Jarvis (2021)

> Artificial intelligence—in which machines are trained to perform jobs and make decisions on their own by studying huge volumes of data ...
> —Adam Satariano (2021)

The most frequent uses of 'AI' in the popular press these days (early 21st century) are to refer to buzzwords such as 'data mining' and 'deep machine learning': training artificial neural networks on many thousands, millions, or even billions (Brown et al., 2020) of data to find or learn patterns in them. This technique has had great success in many aspects of science, medicine, game playing, commerce, computational linguistics, etc. But—despite the implications of the two epigraphs to this section—it is not the only technique of AI.

And it is not without problems:

(1) The "black box" problem is that, in general, the decision-making criteria that a machine-learning program develops on its own from test cases are not always well understood, and certainly not as clear as the workings of a traditional computer program whose code is completely designed by human programmers, and can be read and understood by them. An early statement of the black-box problem is this remark of the philosopher Donald Davidson (1990, p. 2):

> It seems to me that he [Turing] ought also to have worried that it might be possible to rig things so that the circuitry of the nervous system of a person would be reproduced in a digital computer by a method that required no insight into how or why the resulting program of the computer gave the computer what were, or passed for, thoughts.

[3]However, see Halverson et al. 2022 for a different take on this.

As for traditional computer programs, two of the authors of the Logic Theorist said:

> Neither machines nor programs are black boxes; they are artifacts that have been designed, both hardware and software, and we can open them up and look inside. (Newell and Simon, 1976, p. 114)

But contemporary machine-learning programs are, to some extent, "self" designed and *cannot* be "opened up"—they are (like Davidson's nervous-system computer) "opaque" or "black boxes".

(2) The "bias" problem is the related problem that the test cases might have been biased (perhaps unintentionally) and that this might not be evident until the algorithm is deployed:

> And the scariest thing is that many companies have promoted algorithms as a utopia that removes all human flaws. It doesn't. Some neural networks learn from massive amounts of information on the internet—and that information was created by people. That means we are building computer systems that exhibit human bias—against women and people of color, for instance.
> —Cade Metz, as quoted in Ovide 2021

Of course, human behavior also often suffers from the same limitations, so these problems are not necessarily stumbling blocks for AI's success.

When you read about AI in the popular press these days, it is this kind of programming that is usually being referred to. But it is not the only technique of AI. Another is what is sometimes referred to as 'Good Old-Fashioned AI' (GOFAI; Haugeland 1985): the use of logical techniques for reasoning, knowledge representation, planning, problem solving, etc. These techniques are what Newell and Simon had in mind. As a vast oversimplification, we might say that GOFAI techniques tended to ignore neural-network techniques (though it did not ignore machine learning), and current machine-learning techniques based on artificial neural networks tend to ignore logical reasoning and knowledge representation.

Each technique has its strengths. Machine-learning techniques based on artificial neural networks are better at modeling the kind of "implicit" or "tacit" knowledge that we have when we recognize categories or when we do something "instinctively". And GOFAI techniques are better at modeling the kind of "explicit" or "conscious" reasoning that we do.[4]

[4]Cliff Landesman (personal communication, 4 December 2021) pointed out to me that there doesn't seem to be any reason in principle why tacit knowledge couldnt be modeled by symbolic/GOFAI programming. There is, however, one prima facie problem with doing so: Normally, symbolic programming is "glass box" as opposed to "black box". If a symbolic tacit-knowledge pro-

Most likely, a combination of these techniques will be needed if AI is going to succeed. Here is an example where both kinds of cognition might be needed: As a reasonably good driver, I make a left turn "instinctively". That is, I not only don't have to "consciously" think about how to make a left turn, but my "consciously" thinking about how to do it might actually interfere with my doing it successfully. But when I was teaching my son how to drive, I had to observe myself making a left turn so that I could describe what I was doing and then express the steps for doing it in language. (E.g.: Put on left-turn signal; slow down as you approach the turn; turn; speed up as you finish the turn.) My son could then follow that GOFAI algorithm for making a left turn. Eventually, he "internalized" it so that he now does it "instinctively". (And when it's his turn to teach his child how to make a left turn, he'll have to repeat this process.) My *tacit* knowledge of how to make a left turn is, perhaps, best modeled by an artifical neural network, whose workings would have to be "*described*" in a GOFAI program so that they could be analyzed and explained to an automated vehicle that would then have the ability to turn left modeled by one of its artificial neural networks. (For a brief discussion of this problem, see Brachman and Levesque 2022, p. 164. See also Buckner 2024, pp. 297–298.)

The important point for our debate, however, is that all of these techniques are computational (Cummins and Schwarz, 1991). And our question is whether AI—computational cognition—will succeed, not whether one or the other of these (or some combination, or some other computational technique) will be the method that achieves such success. (But recall the discussion in §2.3.3.)

gram were compiled, however, so that it was not changeable, then, arguably, it would become tacit. But then it wouldn't be changeable, as our tacit knowledge can be, at least under some, if not all, circumstances. (A change could be made, but only by re-writing and re-compiling the program.) All of this would need to be investigated and experimented with, but the point remains. Nevertheless, I still think that machine-learning algorithms are closer in spirit, even if not the "best" model.

Further Reading:
With regard to the black-box problem, consider what Turing had to say in response to an objector who didn't see how a computer could do certain intelligent things:

> I've certainly left a great deal to the imagination. If I had given a longer explanation I might have made it seem more certain that what I was describing was feasible, but you would probably feel rather uneasy about it all, and you'd probably exclaim impatiently, 'Well, yes, I see that a machine could do all that, but I wouldn't call it thinking.' As soon as one can see the cause and effect working themselves out in the brain, one regards it as not being thinking, but a sort of *unimaginative donkey-work*. From this point of view one might be tempted to define thinking as consisting of 'those mental processes that we don't understand'. *If this is right then to make a thinking machine is to make one which does interesting things without our really understanding quite how it is done.* (Turing et al., 1952, p. 500, my italics)

Turing's point, however, was probably *not* that "donkey-work" could not be considered to be "thinking". Rather, it was probably that people *mistakenly* believe so; we'll return to this in §4.4.

For a clear statement of the black-box problem (in a fictional context), see Ishiguro 2021, pp. 297–298. For an argument that the black-box problem is not really a problem, see Buckner 2020.

On the bias problem, see Crawford 2016; Savage 2016; Bowles 2019; Metz 2019a,b, 2021; Singer and Metz 2019; Berreby 2020; C.S. Smith 2020. For discussion of both of these ethical problems, see Rapaport 2023a, §17.6.

On implicit vs. explicit knowledge and thinking (sometimes called "Type 1" and "Type 2"), see http://en.wikipedia.org/wiki/Dual_process_theory#Systems and the bibliography at http://www.cse.buffalo.edu/~rapaport/575/rules-connections.html#uncs-cognition. Arguably, Turing Machines do implicit or "unconscious" "Type 1 thinking", because their "programs" are implicit in their construction, whereas Universal Turing Machines do "Type 2 thinking", because they "consciously" execute programs that are explicitly written on their tapes.

For arguments in favor of combining GOFAI and machine-learning techniques, see, especially, Levesque 2017, as well as Garnelo and Shanahan 2019; Landgrebe and Smith 2021b; Seabrook 2019; B.C. Smith 2019; Sablé-Meyer et al. 2021; Brachman and Levesque 2022; Quilty-Dunn et al. 2023. Boden (2018, pp. 86–89) discusses such "hybrid" methodologies, some of which already exist. Note that this is a different use of 'hybrid' from that discussed in §2.3.3. For a possible counter to the argument in Levesque 2017, see Brown et al. 2020, §3.4, which discusses a machine-learning approach to a GOFAI problem. It is also possible that "conscious" GOFAI techniques might help overcome "unconscious" bias by recognizing it and compensating for it.

2.6 Three Goals of AI

> ... if it were to happen [that a computer could write a sonnet,] it would have happened in ways different from how a human source would have obtained a comparable output. In other words, it is not *what* is achieved but *how* it is achieved that matters.
> —Luciano Floridi & Massimo Chiriatti (2020, p. 687)[5]

Are all AI researchers interested in the kind of success that can be measured in terms of human cognition? My colleague Stuart C. Shapiro has argued that there are three distinct goals of AI (Shapiro 1992; see also Rapaport 1998, 2000a, 2003):

1. **AI as advanced computer science or as engineering:**

   > One goal of AI is *to extend the frontiers of what we know how to program* (in order to reach an ultimate goal, perhaps, of computers that are self-programming and that understand natural language) and to do this *by whatever means will do the job*, not necessarily in a "cognitive" fashion. (My former computer-science colleague John Case once told me that AI understood in this way is at the "cutting edge" of computer science. Another former computer-science colleague, Anthony S. Ralston, agreed with Case's topological metaphor, except that instead of describing AI as being at the *cutting* edge, he told me that it was at the *periphery* of computer science!)

2. **AI as computational psychology:**

   > Another goal of AI, expressed perhaps in Boden's definition, is to write programs as *theories or models of human cognitive behavior*. Here, it is important to distinguish the question whether human cognition is comput*able* from the question whether human cognition is comput*ed*. It may turn out that human cognition is computable (and this essay is arguing that it *will* so turn out), yet that the brain is not a computer—that cognition as implemented in humans is not carried out by computational means.

[5]Contrast my remark about music in §2.1.2, above.

3. **AI as computational philosophy:**

> The third goal of AI is to investigate *whether cognition in general* (and not restricted to *human* cognitive behavior) *is computable*, that is, whether it is linguistically expressible as one or more computer programs (more abstractly, as one or more computable functions). (If cognition requires more than one such function, then, presumably, they will be *interacting* functions; see Sidebar B: Computability.) Recall (from §2.4.1) that, if cognition is computable, then it can be implemented in any physical device that computes, not just in (terrestrial) biological organisms.

Our debate, as I frame it, concerns this third goal. As Dietrich et al. 2021, p. 244, put it, "the most interesting kind of AI work [is] the work of creating a geniune artificial mind".

Further Reading:
On AI's three goals, see also Dennett 1979.

On computer programs as scientific theories, see Rapaport 2023a, Ch. 14.

For more on the distinction between being comput*able* and being comput*ed*, see Shagrir 1997, pp. 325f; Rapaport 1998; and Chirimuuta 2021.

For general discussion of whether the brain *is* a computer, see Rapaport 2023a, §9.7.1; see also Shagrir 2006; Piccinini and Bahar 2013; Piccinini 2015, 2020.

2.7 Summary

In this section, we looked at the nature of computer science and of AI, in particular; why AI is "artificial"; and what "intelligence" is. Our discussion can be summarized as the first premise of my argument:

1. AI is the branch of computer science that investigates computational theories of cognition.

3 What Counts as "Success"?

> The first question that will have to be asked is not "Can all *kinds* of thought, logical, poetical, reflective, be imitated by machines?" but "Can *anything* that can be called 'thought' be so imitated and, if so, how much?"
> —M.H.A. Newman (1949)

Clearly, what will count as success for AI will depend in part on the definition of 'AI' and in part on which of its goals we are concerned with. For example, AI as engineering *has* succeeded: Many "apps" and programs use AI techniques (mostly in the guise of machine learning) (Dietrich et al., 2021, Ch. 8, §3). For present purposes, I will only be concerned with AI as trying to develop a computational theory of (human-level) cognition, as discussed in the previous section. So, *AI will succeed to the extent that it is able to show that (and how) all aspects of cognition are computable.*

3.1 Theoretical vs. Practical Success

This is not to say that AI will actually succeed in our lifetime, or even before the heat death of the universe. It is only to say that cognition is comput*able*, not that it *will be* computed or even that (biological) cognition *is* computed by brains.

It might turn out that AI will succeed in this sense of developing a computational theory of cognition, and yet might fail to produce a working computer program that exhibits such full, human-level cognition. To see how this could be, consider the following analogy: It is known that chess is "solvable" in the same sense that tic-tac-toe is: There is a "perfect" way to play these games so that one player can guarantee either a win, a loss, or a draw, depending on how the other player plays. In the case of tic-tac-toe, the algorithm is fairly simple and well known (Zobrist, 2000). A complete "game tree" for tic-tac-toe—a diagram that shows every possible move and every possible reply to each move—can be printed on a single sheet of paper (https://xkcd.com/832/). But chess is, to put it mildly, vastly more complex than tic-tac-toe: A complete game tree for chess would require more space and time than there are in the universe (Zobrist 2000, p. 367; Rapaport 2023a, Chs. 3, 7). So, as a *practical* matter, even though there is a computable theory of how to play perfect chess, no computer will be able to do it. Perfect chess is mathematically possible but practically *intractable*.

This is not, of course, to say that computers won't ever be able to play *nearly* perfect chess; in fact, they do![6] But they do this using *heuristic* algorithms: A *heuristic for problem p* can be defined as an algorithm for a *different* (but related) problem p', where the solution to p' is "good enough" as a solution to p (Rapaport, 1998, p. 406). Being "good enough" is, of course, a subjective notion; Oommen and Rueda (2005, p. 1) call the "good enough" solution "a *sub-optimal* solution that, hopefully, is arbitrarily close to the *optimal*." The idea is related to Herbert Simon's notion of "bounded rationality": We might not be able to solve p because

[6]https://en.wikipedia.org/wiki/Computer_chess
and https://en.wikipedia.org/wiki/Deep_Blue_versus_Garry_Kasparov

of limitations in space, time, or knowledge, but we might be able to solve p' algorithmically within the required spatio-temporal-epistemic limits. And if the *algorithmic* solution to p' gets us closer to a solution to p, then it is a *heuristic* solution to p. But it is still an algorithm. A classic case of this is the Traveling Salesperson Problem, an *NP*-complete problem that software like Google Maps solves special cases of every day (even if their solutions are only “satisficing” ones. (On *NP* problems, see Sidebar B.)

So it might be the case that, even if AI succeeds in its goal of showing how all aspects of cognition are computable, we might not be able to build a computer that can do it, roughly on the grounds that full, human-level cognition is probably at least as complex as chess. The best we might be able to hope for is a *nearly* perfect “intelligent” robot. We’ll return to this in §6.

Further Reading:
For an interesting take on different ways in which AI can “succeed”, see Aaronson and Barak 2023.

Two important surveys of meanings of the term ‘heuristic’ are Romanycia and Pelletier 1985; Chow 2015. Simon and Newell 1958 distinguishes *algorithmic* problem solving of “well-structured” problems from *heuristic* problem solving of “ill-structured” problems. Other discussions of heuristics include the classic Polya 1957, and Newell and Simon 1976; Korf 1992; S.C. Shapiro 1992; Findler 1993. A symposium on heuristics in AI appeared in *Minds and Machines* 5(4) (November 1995): 467–512,, https://link.springer.com/journal/11023/volumes-and-issues/5-4. Although not about heuristics, Thagard 2007 presents a theory of “approximate” truth that bears a close resemblance to the idea that a heuristic is an algorithm that computes an approximately correct answer.

‘Satisficing’ means “finding optimal solutions for a simplified world, or finding satisfactory solutions for a more realistic world” (Simon, 1978, p. 350); see also Simon 1959, 1996a; Griffiths 2020.

On intractable problems and NP-completeness, see Sidebar B and Austin 1983; Cook 1983; Mycielski 1983; Fortnow 2009, 2013, 2022; Walsh 2014. On the Traveling Salesperson Problem, see Lipton 2020; Fortnow 2022.

3.2 Measuring and Evaluating Success

3.2.1 Measuring Devices

In order to decide if AI will succeed (or eventually will have succeeded), we need to know how to *measure* such success and how to *evaluate* it. That is, we need some kind of measuring "device" for testing an AI theory, and we need some way of determining if the theory "passes" the test.

For any particular aspect of cognition, a method can probably be developed for testing that aspect. For example, suppose we are interested in the cognitive ability to identify pictures of cats (Le et al., 2012; Markoff, 2012). We might first develop a computational theory of how to do this, using GOFAI or machine-learning techniques (or a combination of them, or some other computational technique not yet imagined). To see if the theory (or the program) works ("succeeds"), we might show it pictures of cats and non-cats, and see if it identifies the cats as cats and does not identify the non-cats as cats. That would be the *measuring device*. To *evaluate* the theory (or the program), we would want some statistics about the number of correct answers it gets. Roughly, the more correct answers and the fewer incorrect ones, the more successful it is.

Does it have to identify *every* picture correctly in order to be considered completely successful? Well, that would certainly be nice, but if our goal is to *match* human-level cognition, such perfection might not be necessary. After all, it's unlikely that every human would always and only make correct identifications: There might be "borderline" cases: Is a baby tiger a cat, or do only "housecats" count? What about cartoon drawings of cats? Or there might be blurry pictures, partially occluded cats, and so on. So one way to evaluate this very small aspect of cognition might be to take a large, random sample of humans, show them the set of pictures, measure their success, and then use that as a "gold standard" against which to evaluate the computational theory. Note that a "successful" computer program might come close to, match, or even exceed human performance.

But now what about full, human-level cognition—not only identifying cats, but using language, solving problems, perceiving ... the whole works? How might we test that?

3.2.2 The Turing Test

> Alan Turing's test has been attacked by some of the sharpest minds in the business. To date, however, it stands unrefuted. In fact, it is the only viable proposal on the table for testing whether a computer is capable of thought.
> —Jack Copeland (2017b, p. 275)

The most famous such test of what some call 'artificial general intelligence' was proposed by Turing in 1950. A student in one of my introductory computer science classes, having confused the Turing Machine mathematical model of computation with what is now known as the Turing Test for AI, wrote on his final exam that "a problem is computable if a computer can convince you it is". Although this is not(!) a correct definition of 'computable', with a few small changes it becomes a not-unreasonable statement of the Turing Test: A computer is *cognitive* if it can convince you it is.

The point of the Turing Test is for a human to judge whether an entity can think, i.e., whether its cognitive behavior is indistinguishable from that of a human. Just as Turing's most important paper (Turing, 1936) never mentions a "Turing Machine", his second most important paper (Turing, 1950) never mentions a "Turing Test". Instead, he introduces a parlor game that he calls the "Imitation Game", which you can actually play (try it!).

The Imitation Game consists of three players: A man, a woman, and an interrogator who might be either a man or a woman. The three players are placed in separate rooms, so that they cannot see each other, and they communicate only by means of what we would now call 'texting', so that they cannot hear each other. The reason that they are not allowed to see or hear each other is that the point of the game is for the interrogator to determine which room has the man and which room has the woman. To make things interesting, the woman is supposed to tell the truth in order to convince the interrogator that she really is the woman, but the man is supposed to convince the interrogator that he (the man) is the woman, so he may occasionally have to lie. The man wins if he convinces (fools) the interrogator that he is the woman; the woman wins if she convinces the interrogator that she is the woman. If the man wins, then he is said to have passed the test.

Turing suggested that "an average interrogator will not have more than 70 per cent. chance of making the right identification after five minutes of questioning" (Turing, 1950, p. 442). But the frequency and amount of time are to some extent irrelevant. One could conduct a series of imitation games and calculate appropriate statistics on how likely an interrogator is to make a correct determination after different periods of time.

What does this game have to do with whether AI will succeed? In popular parlance, for AI to succeed is for computers to think. What has come to be known as the Turing Test makes one small change in the Imitation Game:

> We now ask the question, "What will happen when a machine takes the part of [the man] in this game?" Will the interrogator decide wrongly as often when the game is played like this as ... when the game is played between a man and a woman? These questions replace our original, "Can machines think?" (Turing, 1950, p. 434)

There is an ambiguity in Turing's essay, concerning whether the computer is supposed to convince the interrogator (a) that it is a man, or (b) that it is a man who is trying to convince the interrogator that he is a woman, or (c) that it is a human. Usually, the Turing Test is taken, more simply and less ambiguously, to consist of a set-up in which a computer, a human, and a human interrogator are located in three different rooms, communicating over a texting interface, and in which both the human and the computer are supposed to convince the interrogator that each is a *human*. To the extent that the computer convinces the interrogator that it is a human (under the same criteria for successful convincing that obtain in the original imitation game), the computer is said to have passed the Turing Test.

An even simpler version consists merely of two players: a human interrogator and someone or something (a human or a computer) in two separate, text-interfaced rooms. If a computer convinces the interrogator that it is a human, then it passes the Turing Test.

Often, the Turing Test is considered to be a *behavioral* test of thinking (or intelligence, or cognition). That is, a computer that passes a Turing Test may act and behave *as if* it thinks (or is intelligent, or exhibits cognition), but it might "merely" be executing a "donkey-work" computer program and thus not "really" thinking, or not "really" being intelligent. (Recall the quotation from Turing in the Further Reading at the end of §2.5.) The most well-known objection along these lines is Searle's Chinese Room Argument (see Sidebar A).

I won't get into these issues here. I do want to point out two things, however. First, a Turing Test—or even a more elaborate one for a robot, which might test not merely contemplative "thinking" but also "intelligent" *action*, as in Stevan Harnad's "Total Turing Test" (1991) or Paul Schweizer's "Truly Total Turing Test" (1998)—is pretty much the same kind of test that we had for cat-identification. Moreover, it is exactly the way that we would determine if an extra-terrestrial who visited Earth was "intelligent": We would see how it behaved. If it behaved in ways that we would normally take to be "intelligent", we would have no reason not to think that the extra-terrestrial was "intelligent" (recall the quotation from Copeland in §2.3.1). In fact, it is exactly the way that *I* determine if *you* are "intelligent" (and vice versa)!

Second, here is Turing's answer to the question that has now replaced "Can machines think?":

> I believe that at the end of the century [which would have been by the year 2000] *the use of words* and *general educated opinion* will have *altered* so much that one will be able to speak of machines thinking without expecting to be contradicted. (Turing, 1950, p. 442, my italics)

This is a very interestingly subtle *variation* on the behavioral theme. He did not say that machines *will* think, nor did he say that machines will *behave as if* they think, nor did he say (as many take him to have said) that "intelligent" *behavior* is *all* that matters. Rather, he said—I repeat—that "the use of words and general educated opinion will have altered so much that one will be able to speak of machines thinking without expecting to be contradicted". One reason that "machines that think" can be considered a contradiction is that one definition of 'machine' and its cognate 'mechanical' *excludes* "thinking". [7] Shanker 1987, p. 616 quotes the philosopher Ludwig Wittgenstein as saying that

> the sentence, "A machine thinks (perceives, wishes)": seems somehow nonsensical. It is as though we had asked "Has the number 3 a colour?" (Wittgenstein, 1934, p.47)[8]

So, Turing's point, arguably, is that whatever it is that a successful AI-programmed computer or robot does will be *called* 'thinking', and what we *define* 'thinking' to be will include such behavior. (We'll come back to this in §3.2.3.)

This is not unreasonable. Such changes in our language and our scientific theories have already occurred in at least two other situations: First, "flying" is no longer something that only certain insects, birds, and mammals do. Airplanes also fly (our use of words has changed). And our physical theories of flight are no longer limited to flapping wings but are based on theories of aerodynamics that encompass both animal and machine flight (general educated opinion has changed).

Second, "computers" are no longer humans who compute (as they were in an 1892 ad in *The New York Times* for a computer[9] or when Turing wrote his 1936 paper); certain machines are also computers. The use of words has changed; in fact, it has changed so much that 'computer' now principally refers to computing *machines*, and it is only with great difficulty that a 21st-century reader can understand Turing's 1936 paper that uses pronouns like 'he' to refer to computers! And our theories of computation encompass both what humans do and what computers do (general educated opinion has changed) (Rapaport 2000b; Rapaport 2023a, Ch. 18).

[7] See the *Oxford English Dictionary*'s definitions of 'machine' at http://www.oed.com/view/Entry/111850, senses IV.6.b and V.8.b, and of its cognate 'mechanical' at http://www.oed.com/view/Entry/115544, sense A.7. See also Sieg 2008b, pp.527, fn.1, 574.

[8] But cf. Proudfoot 2024 for commentary on this passage.

[9] https://danwin.com/2013/02/the-first-mention-of-computer-in-the-new-york-times/

Further Reading:
On "artificial general intelligence", see Goertzel and Pennachin 2007 and https://en.wikipedia.org/wiki/Artificial_general_intelligence. For a pessimistic view of it, see Landgrebe and Smith 2019. Dietrich et al. 2021, pp. 121–123, distinguishes between "monointelligences" that can only do one cognitive task (albeit extremely well, such as playing winning chess) and more general intelligence that can do many (or all) cognitive tasks.

Argamon et al. 2003 provides evidence that women *can* be distinguished linguistically from men on the basis of their writing style.

The differences between the various versions of the Turing Test are discussed by French 2000; Piccinini 2000; Rapaport 2006c. Turing (1952) argued against the two-player version, and Donald Davidson (1990, p. 6) says that it "is not the same test as Turing's"; for discussion, see Proudfoot 2017a, p. 290. For other variations, see Sterrett 2020. For general discussion of the Turing Test and objections to it, see especially Akman and Blackburn 2000; Moor 2003; Shieber 2004; Damassino and Novelli 2020. I discuss it further in Rapaport 2000b, 2006c; and Rapaport 2023a, Ch. 18. Descartes (1637) anticipated the Turing Test; for discussion, see Gunderson 1964, p. 198.

Davidson (1990, pp. 8–9) also advocates for a test that goes beyond verbal behavior. On Turing-like tests for non-verbal intelligence, see Crosby 2020. On more practical Turing-like tests, see the Further Reading at the end of §2.4.1.

Minsky was slightly more optimistic than Turing on the date for AI "success", writing in his computability-theory textbook in 1967 that "Within a generation, I am convinced, few compartments of intellect will remain outside the machine's realm—the problems of creating 'artificial intelligence' will be substantially solved." Using 25 years as the measure of "a generation", that would have been 1992. Turing (1952) later extended his end-of-the-century target date to "at least 100 years", i.e., no earlier than 2052, so we still have a few years to go!

Proudfoot 2017b argues that the Turing Test is *not* a behavioral test and that Turing was *not* a behaviorist. Recall, too, her claim that Turing understood "intelligence" as in the eye of the beholder; see the Further Reading box at the end of §2.4.1.

The phrase "use of words" appears in a similar context in Jefferson 1949, p. 1110, which may have inspired Turing's usage. One way that our use of words can change is by "metaphorical extension" (cf. Lakoff and Johnson 1980a,b; Strawson 2005, p.43; Xu et al. 2017; Ramiro et al. 2018). For discussion in the context of the Turing Test, see Rapaport 2000b, §2; Rapaport 2023a, §18.3.3; Miłkowski 2022, p. 6 of 20.

3.2.3 AI's Successes

This view of Turing's about "the use of words and general educated opinion" suggests that AI might "succeed" *by definition*! That's not exactly the kind of success that our current debate is about. Moreover, AI techniques have been developed for almost all aspects of cognition, even if these techniques are not "complete" or "perfect". There are AI programs that can generate and (at least apparently) understand (some aspects of) natural language (Brown et al. 2020; but see Mitchell 2021); programs that can perceive and identify (some) objects (Le et al., 2012); programs that can produce "creative" writing, painting, and music (see Boden 1990; Ebcioğlu 1990; but contrast Kirkpatrick 2023); programs that can solve problems; programs that can prove mathematical theorems (even some that humans were not able to prove); programs that can play winning checkers, chess, Go, and backgammon (see, e.g., Schaeffer et al. 2007; Silver et al. 2018; Halina 2021);[10] and programs that can learn. There are even computational theories of (aspects of) emotion and consciousness.

Granted, in most areas, the AI programs that can do one of these things cannot do the others. What's needed is "artificial *general* intelligence", understood as an integration of all of these features, as well as the ability of a cognitive agent to connect all of its understanding—to have a holistic semantic network as part of its memory. As Stuart C. Shapiro (1992, pp. 56–57) has noted, most of these aspects of cognition are "AI-complete", in the sense that a solution to any one of them will require (or yield) a solution to all of them. Even so, AI as a research field has been reasonably successful (for good surveys, see Edelman 2008; Boden 2018; or any introductory AI textbook, such as Russell and Norvig 2020). The question is *how* successful it can be. Might AI fail in its goal of showing that *all* of cognition is computable? Might it be the case, as Stuart Dreyfus (Dreyfus and Dreyfus, 1986, p. 10) has said, that "claims and hopes for progress in models for making computers intelligent are like the belief that someone climbing a tree is making progress toward reaching the moon"? We need to show that we are not merely climbing a tree.[11]

[10]But see Marcus 2023 for an important caveat about a Go program!

[11]There is an interesting similarity between the AI project as I have presented it and one approach to the foundations of mathematics:

> Kronecker admitted as objects of analysis only natural numbers[,] and constructed from them, in now well-known ways, integers, rationals, and even algebraic reals. ... Clearly, this procedure is strictly arithmetic, and Kronecker believed that analysis could be re-obtained by following it. ... Such a program is not chimerical, as mathematical work during the last two decades has established that a good deal of analysis and algebra can be done in conservative extensions of primitive recursive arithmetic. (Sieg, 1999, p. 4)

Further Reading:
A good survey of AI's successes is Brachman and Levesque 2022, Ch. 3. On computational theories of emotion, see Sloman and Croucher 1981; Picard 1997; Minsky 2006; Thagard 2006. On computational theories of consciousness, see the bibliography at https://cse.buffalo.edu/~rapaport/719/csnessrdgs.html

On holistic semantic networks, see Rapaport 1988a, 1995, 2000b, 2002, 2006a, 2012. On the need for an integration of many features, see the discussion of "monointelligences" vs. "open-ended" intelligences in Dietrich et al. 2021, pp. 117, 121–123, and their example of a holistic robotic pencil sharpener (pp. 101–105). On the history of the term "AI-complete", see https://en.wikipedia.org/wiki/AI-complete and Yampolskiy 2012.

3.3 Summary

Thus, we can take as further premises of my argument:

2. AI's success can be measured in terms of how much of cognition can be shown to be computable.

3. AI will completely succeed if it can show that *all* (or *all essential*, or *nearly all*) aspects of cognition are computable (or heuristically computable).

4. There are many essential aspects of cognition that have been shown to be computable.

Therefore,

5. AI has partially succeeded.

It is, of course, an open question whether AI will completely succeed (else we would not be having this debate). AI is an ongoing science. Are there any reasons to think that AI will *not* completely succeed—that it will fail?

Just as this mathematical program's goal is to see how much of mathematics can be constructed from arithemtic, so the goal of AI is to see how much of cognition is computable. And just as the mathematical project is not "chimerical" due to its successes, neither is the AI project, due to *its* successes.

Alternatively, my approach is akin to that of Aaronson 2016, p. 198; to paraphrase him: "I want to see *how far I can get* in thinking about [cognition] in a [computational] way. 'Resolving' the millennia-old ... debate [about whether machines can think] isn't even on the table! The most I can hope for, if I'm lucky, is to construct a model whose strengths and weaknesses help to move the debate slightly forward."

4 What Counts as 'Failure'?

... there's a big difference between proving that something is false and failing to prove that it's true ....
—Margaret Boden (2018, p. 32)

There are at least four reasons why some have thought that AI won't succeed: I will call them "practical" reasons, "biological" reasons, "vitalistic" reasons, and "mathematical" reasons. I don't think that any of them are good reasons, but none of them should be dismissed out of hand, and we can learn a lot from some of them.

Note that none of them are arguments against the *possibility* of AI or against the possibility of AI succeeding in *some* areas of cognition. As we just saw, AI has already *partially* succeeded.

4.1 Practical Arguments against AI's Success

... there is no obstacle *in principle* to human-level AI intelligence ....
The question ... is whether this is likely *in practice*.
—Margaret Boden (2018, p. 136)

One obvious but uninteresting kind of practical reason why AI might fail is that there might not be enough funding for it to succeed. Other such "defeaters" (as Chalmers 2010, p. 7, calls them) include such things as "disasters, disinclination, and active prevention". But the kinds of practical arguments against AI's success that I have in mind are ones that point to what appear (to their authors at least) to be insurmountable difficulties in particular areas of AI: AI will fail because it's too hard.

One of the earliest of these was the claim by philosopher Hubert Dreyfus (1965) that computers would not be able to play chess well, roughly on the grounds that it was too difficult. Claims of this sort have often been taken by AI researchers as challenges to be overcome—as new goals for their research—rather than as roadblocks to their methodology. And, of course, the chess challenge was indeed overcome. (Dreyfus had more serious objections, too, as we'll see in §4.2.)

A more recent example of a practical argument against AI is a lengthy discussion by a computer scientist (Jobst Landgrebe) and a philosopher (Barry Smith) about the limitations of AI (Landgrebe and Smith 2019; see also Landgrebe and Smith 2021b,a). They argue in detail that human conversational behavior is simply too difficult to achieve, citing many examples of such behavior that are beyond the capabilities of current computational theories.

But their arguments—and all arguments of this type[12]—take the following form:

1. Here are all of the very difficult things that a computer would have to be able to do in order to pass a Turing-like Test.
2. Here are all of the currently known mathematical and computational methods for solving problems like those.
3. None of them can do the job.
4. ∴ The job can't be done.

I have no problems with the first two premises, which merely present unobjectionable data. And I am willing (for the sake of the argument) to accept the third premise. But this sort of argument as it stands is invalid, arguing from an "is not" to a "cannot".

At best, there's a missing premise, namely:

> 3.5. There *cannot* be any *other* methods besides those cited in premise 2.

Note that premise 2 only talks about "currently known" methods. Before Newton's and Leibniz's time, physicists might have drawn the same conclusion about understanding gravity or planetary motion; but Newton and Leibniz (independently) invented calculus, a hitherto unknown mathematical method that enabled an understanding of these phenomena. There is no reason to believe missing premise 3.5.

One question that Landgrebe and Smith leave unanswered is this: What is it about humans (and other animals) that allows them to "overcome" these limitations? We'll look at some possibilities in §§4.2 and 4.4. But the main problem with such arguments is that their real claim is that human cognition is not scientifically understandable—that no theories will allow us to understand it or, therefore, to re-create it. That's possible, of course; but it is anti-scientific at best, and merely autobiographical at worst ("I can't imagine how to solve this problem; therefore, it cannot be solved.").

A nice example of the differing attitudes that can be taken to such difficulties is the contrast between Landgrebe and Smith (2019), on the one hand, and computer scientist and philosopher Brian Cantwell Smith (2019), on the other. Landgrebe and Barry Smith argue that, because there is no *known* way for certain aspects of cognition (in particular, human language use) to be handled mathematically, they *cannot* be handled computationally. Brian Cantwell Smith argues that *new methods* may have to be developed before such aspects can be handled. From the

[12]For another, see Fetzer 2011 and my reply in Rapaport 2012.

same starting point of agreed-upon difficulties, Landgrebe and B. Smith conclude pessimistically that they cannot be overcome, while B.C. Smith concludes optimistically that—with a lot more work and innovations—they can (or, at least, that they will have to be). Barry Smith (personal communication, 9/29/2020) says that Brian Cantwell Smith's position is "analogous to defenders of the possibility of perpetual motion machines". But there are independent arguments of the impossibility of perpetual motion machines; whether there are independent arguments for the impossibility of AI is precisely the issue we are debating.

I grant that there are many deep and difficult problems about the nature of cognition that are yet to be solved. I will also grant that, given the limits of human ingenuity as well as limits of time (and funding), they may never in fact be solved. That is, they may be unsolvable in practice, but that does not mean that they are unsolvable in principle.

4.2 Biological Arguments against AI's Success

The kinds of "biological" arguments that I have in mind are ones that claim that there is something special about humans that cannot be duplicated (or simulated) by computers—that human cognition is not susceptible of purely computational or mathematically formal analysis, but depends in an essential way on human biology.

4.2.1 Hubert Dreyfus's Argument

One kind of argument along these lines was due to Dreyfus (1992), saying that human cognition is based on largely unconscious human abilities to understand the full context of situations and to act on the basis of these kinds of tacit knowledge. This was accompanied by a claim that such knowledge and the ability to use it could not be modeled by computational methods. The computational methods that he had in mind were the dominant GOFAI ones of the time: rigorous following of explicit logical rules.

To some extent, he may have been right about those particular methods. But machine-learning methods based on artificial neural networks have gone a long way to show how computational methods of a different type can in fact model unconscious or tacit knowledge of how to do things "intelligently"—knowledge that we cannot (easily) express in language. Indeed, this ineffability or "opacity" is one of the problems of such methods: Even their programmers typically do not fully understand the actual patterns that such programs discover or that they base their behavior on. What's problematic about this black-box inability to explain what they are doing is the bias problem mentioned in §2.5. But they do arguably behave in exactly the kinds of ways that Dreyfus argued was necessary for successful cognition but that he thought was impossible. Moreover, as I noted in §2.5, both GOFAI methods and machine-learning methods have limitations, but it may turn out that one method's limitations may be the other's strength, so that a combination of methods may be stronger than either alone.

4.2.2 John Searle's Argument

Another kind of argument from biology can be found in Searle's Chinese Room Argument (see Sidebar A), where he argues that "only something that has the same causal powers as brains can have intentionality" (Searle, 1980, p. 423).[13] By 'intentionality' here, Searle means "cognition" more generally, and "understanding" in particular. (Recall our discussion of this term in §2.3.1.)

What are these causal powers? After all, if they turn out to be something that can be computationally implemented, then computers can have them (which Searle thinks they cannot). He says that these causal powers are due to the fact that "I am a certain sort of organism with a certain biological (i.e. chemical and physical) *structure*" (Searle, 1980, p. 422, my italics). That narrows down the nature of these causal powers a little bit. If we could figure out what this biological *structure* is, and if we could figure out how to implement that structure computationally, then we should be able to get computers to understand. Admittedly, those are big "if"s, but they are worth trying to satisfy.

So, what is this biological structure? Before we see what Searle says about it, let's think for a moment about what a "structure" is. What is the "structure" of the brain? One overly simplistic, but plausible, answer is that the brain is a network of neurons, and the way those neurons are organized is its "structure". Presumably, if you made a model of the brain using string to model the neurons, then, if the strings were arranged in the same way that the neurons were, we could say that the model had the same "structure" as the brain. Of course, string is static (it doesn't do anything), and neurons are dynamic (they can be thought of as mini-computers; see McCulloch and Pitts 1943), so structure alone won't suffice.

However, Searle doesn't think that even structure *plus* the ability to do something is enough: He says that a simulated human brain "made entirely of . . . millions (or billions) of old beer cans that are rigged up to levers and powered by windmills" would not really exhibit cognition even though it appeared to (Searle, 1982). Cognition must (also) be *biological*, according to Searle. That is, it must be made of the right stuff.

But now consider what Searle is saying: Only biological systems have the requisite causal properties to produce cognition. So we're back at our first question:

[13]This section is adapted from Rapaport 2023a, Ch. 18.

What are those causal properties? According to Searle, they are the ones that are "causally capable of producing perception, action, understanding, learning, and other intentional [i.e., cognitive] phenomena" (Searle, 1980, p. 422). Again: What are the causal properties that produce cognition? They are the ones that produce cognition! That's not a very helpful answer, even if it's true that *human* cognition is biological.

The philosopher and computer scientist Aaron Sloman (2020b) has suggested that Turing may have thought something similar when, in Turing's 1950 paper on the Turing Test, he wrote:

> In the nervous system chemical phenomena are at least as important as electrical. (Turing, 1950, p. 439)

Sloman's interpretation of this passage is that human (more generally, biological) cognition might not be solely computational, but might also depend on its physical (including chemical and biological) implementation. One of Turing's philosophical sparring partners, Geoffrey Jefferson (1949, p. 1108), made a similar observation. Gualtiero Piccinini (2020) argues that the neural implementation level is essential to an understanding of cognition—i.e., *human* cognition. And Landgrebe and Smith argue that the full details of neural implementation make their computationl modeling impossible.[14] If that's the case, then a purely computational theory of cognition would inevitably miss out on something.

Or would it? Presumably, other (non-biological) kinds of cognition (such as those that might be exhibited by extra-terrestrials or robots) might also depend on *their* physical (including chemical and exo-biological) implementations. But all this suggests is that any non-human re-implementation of cognition will have to take the non-human physical implementation level into consideration. Alternatively, it might turn out that any "essential" chemical or physical component of (human) cognition could itself be modeled computationally (see, e.g., Boden 2018, pp. 84–86; Halverson et al. 2022). But there would still be a level of abstraction that would make them all count as "cognitive". And *that* level—a *psychological* level—might be computational.

The AI project is different from just understanding human cognition; it is to understand cognition more generally. The goal of AI isn't (or shouldn't be) to precisely *duplicate* human cognition, but to implement cognition in non-human devices, i.e., to develop a computational *theory* of cognition that is independent of, but that includes, human cognition. After all, we wouldn't expect Martian cognition to duplicate human cognition, any more that we expect non-human *animal* cognition to duplicate human cognition.

[14]Piccinini 2020, p.190 makes a similar point.

But I hold that the goal of AI is to investigate whether cognition (restricted to human-level, if you want) is computable. And that can be investigated and can even result in a positive answer even if human-level cognition is necessarily biological, in the sense that it is the result of the electrochemistry of the brain. *The issue is whether cognition of the kind that humans have can be produced in another way, by computation.* As I indicated above, brains might do things differently from how computers do them, relying on their chemistry, perhaps. But the interesting AI question is whether computers can do them at all, i.e., whether those tasks are computable. And by computation is meant Turing computation. So, if some kind of computation that goes "beyond" Turing Machines (so-called "hypercomputation") is needed, that's a non-starter (see Sidebar B.1). If probabilistic or neural-network computation is needed, that might depend on whether such computation can be simulated (to any desired degree of accuracy) on a Turing Machine.

4.3 Landgrebe and Smith's Argument

A biological argument has been proposed by Landgrebe and Smith (2023) that is a bit more explicit about the kind of causality involved. They argue that the human brain is a living organism, not a machine. Living organisms are a kind of "complex system" that "produce energy-storing biomolecules … which allow them to survive and to reproduce …" (p. 196). Moreover, "there is no way to model the behaviour of a complex system with the accuracy necessary to support sound technical applications" (p. 195). Their notion of "complex system" comes from Thurner et al. (2018): It is one "whose states change as a result of interactions and whose interactions change concurrently as a result of states. Due to this chicken-egg-type problem, complex systems show an extremely rich spectrum of behaviour" (Thurner et al., 2018, p. v). And a "driven" complex system is one that "is (exogenously) driven away from its equilibrium states" (Thurner et al., 2018, p. 9).

Presumably, machines either don't or cannot have the features of energy production, survival, reproduction, and drivenness. Perhaps they *don't* have them; it is not obvious why they *cannot*. But even if they cannot, why would they have to? It is also not obvious why these abilities would be necessary for cognition. Landgrebe and Smith argue that there is no *currently known* way to model complex systems, but that does not mean that they *cannot* be modeled or, more to the point, that they would *have* to be modeled in order to achieve computational cognition. Even if *our* (human-level) intelligence might require a computationally intractable (or even uncomputable) complex system, there could be other kinds of intelligence (including intelligence that is "heuristically close" (see §3.1) to human-level) that *are* computable.[15]

In any case, the experts that Landgrebe and Smith cite on driven complex systems also say "There are those who say that complex systems will never be understood or that, by their very nature, they are incomprehensible. This book will demonstrate that such statements are incorrect. … [C]omplex systems are algorithmic" (Thurner et al., 2018, pp. vi, 7).

4.4 Vitalistic Arguments against AI's Success

> We still haven't arrived at 'real' understanding in robots, but we are getting closer. That, at least, is the conviction of those of us inspired by Turing's insight. The trickle-down theorists are sure in their bones that no amount of further building will ever get us to the real thing. They think that a Cartesian *res cogitans*, a thinking thing, cannot be constructed out of Turing's building blocks.
> —Daniel C. Dennett (2013b, p. 573)

> The development of Turing's universal machine involved the stripping out of all but operational (mechanical) features of calculation. A startling feature of the Turing machine was the reduction of calculation, for which the *sine qua non* had until then been human intelligence, to a series of unthinking tasks.
> —Doron D. Swade (2017, p. 257)

> The real point about AI is that we are increasingly decoupling the ability to solve a problem effectively—as regards the final goal—from any need to be intelligent to do so.
> —Luciano Floridi & Massimo Chiriatti (2020, p. 687)

Vitalism is the theory, now almost universally rejected by scientists, that life is not "merely" a natural, but very complex, kind of organic chemistry, but that it involves, in an essential way, something "supernatural"—a "vitalistic spirit"—that infuses the chemistry with life (see, e.g., Zimmer 2021). The idea behind this is that some people cannot understand how living matter can emerge from non-living chemicals. Yet it does.

In a similar vein, some people argue "that human thought has some kind of magical quality which resists rational description" (LaForte et al., 1998, p. 285), that cognition cannot emerge from physical matter, that "mind" is completely distinct from "matter": As a humorous "Short Cut to Metaphysics" put it in an

[15] On this kind of argument, compare the discussion in Oppy and Dowe 2020, §2.7, of Turing's (1950, pp. 451–452) response to the "Argument from Continuity in the Nervous System".

1855 issue of *Punch*, "What is Matter?—Never mind./What is Mind?—No matter."[16] Descartes (1641) is perhaps the most famous supporter of this position, having argued that mind and body (in more contemporary terminology, mind and brain) were two different kinds of substance that (somehow, mysteriously) interacted. Almost no one now believes this (though some well-known scientists and philosophers do, e.g., Popper and Eccles 1977). Those who view cognition as "reducible" to physics, chemistry, and biology don't necessarily need to argue that every statement about cognition can or must eventually be expressed in purely physical (chemical, biological) terms, though some do.

Just as it is difficult to see how life is nothing but chemistry, it is difficult (some would say impossible) to see how cognition could come about as the result of the kind of "mere" symbol manipulation that is involved in programming a computer. Compare the following from an evolutionary biologist . . .

> The possibility of the deliberate creation of living organisms from elementary materials that are not themselves alive has engaged the human imagination for a very long time. (Lewontin, 2014, p. 22)

. . . to this paraphrase:

> The possibility of the deliberate creation of intelligent behavior from elementary operations that are not themselves intelligent has engaged the human imagination for a very long time.

The most famous expression of this is due to the 18th-century philosopher Gottfried Wilhelm Leibniz:

> Imagine there were a machine whose *structure* produced thought, feeling, and perception; we can conceive of its being enlarged while maintaining the same relative proportions [among its parts], so that we could walk into it as we can walk into a mill. Suppose we do walk into it; all we would find there are cogs and levers and so on pushing one another, and never anything to account for a perception. (Leibniz, 1714, §17, translator's bracketed interpolation)

Leibniz was looking at things from the bottom up, but a top-down approach can make it more plausible. However, one must be cautious: An infamous top-down approach is the theory of the "homunculus" (a Latin word meaning "little man"; plural = 'homunculi'): In the philosophy of mind and perception, a possible explanation of how we see is that light enters our eyes, an image is formed on the retina, and a "little man" inside our brain sees it. The problem with this, of course, is that

[16] https://quoteinvestigator.com/2012/12/04/those-who-mind/#note-4956-1

it doesn't explain how the *homunculus* sees. Postulating a second homunculus in the first homunculus's brain just postpones the solution.[17]

Dennett offers a recursive alternative (see Sidebar D: Recursion) that avoids this infinite regress, with the base case being something that can just say 'yes' or 'no' when asked:[18]

> The AI programmer begins with an Intentionally characterized problem, and thus frankly views the computer anthropomorphically: if he [sic] solves the problem he will say he has designed a computer that can understand questions in English. His first and highest level of design breaks the computer down into subsystems, each of which is given Intentionally characterized tasks; he composes a flow chart of evaluators, rememberers, discriminators, overseers and the like. These are homunculi with a vengeance; the highest level design breaks the computer down into a committee or army of intelligent homunculi with purposes, information and strategies. Each homunculus in turn is analysed into smaller homunculi, but more important into less clever homunculi. When the level is reached where the homunculi are no more than adders and subtracters, by the time they need only the intelligence to pick the larger of two numbers when directed to, they have been reduced to functionaries 'who can be replaced by a machine'. The aid to comprehension of anthropomorphizing the elements just about lapses at this point, and a mechanistic view of the proceedings becomes workable and comprehensible. (Dennett, 1975, pp. 178–179)

As with so much else in computer science and AI, Turing anticipated this:

> The 'skin of an onion' analogy is also helpful. In considering the functions of the mind or the brain we find certain operations which we can explain in purely mechanical terms. This we say does not correspond to the real mind: it is a sort of skin which we must strip off if we are to find the real mind. But then in what remains we find a further skin to be stripped off, and so on. Proceeding in this way do we ever come to the 'real' mind, or do we eventually come to the skin which has nothing in it? In the latter case the whole mind is mechanical. (Turing, 1950, pp. 454–455)[19]

[17]https://en.wikipedia.org/wiki/Homunculus_argument

[18]In this passage, 'Intentionally' with a capital-T refers to Dennett's "Intentional stance"; recall §2.3.1.

[19]In fairness, however, I should point out that Turing continues this passage by saying of the mechanical mind, "It would not be a discrete-state machine, however. We have discussed this." Presumably, he is referring to his reply to the "Argument from Continuity in the Nervous System", where he says that "The nervous system is certainly not a discrete-state machine" (Turing, 1950, p. 451).

But another approach to Leibniz's puzzle is to bite the bullet. Dennett first noted this in the context of Darwin's theory of evolution, citing a critic of Darwin who attempted to show that Darwin's theory was nonsense:

> In the theory with which we have to deal, Absolute Ignorance is the artificer; so that we may enunciate as the fundamental principle of the whole system, that, IN ORDER TO MAKE A PERFECT AND BEAUTIFUL MACHINE, IT IS NOT REQUISITE TO KNOW HOW TO MAKE IT. This proposition will be found, on careful examination, to express, in condensed form, the essential purport of the Theory, and to express in a few words all Mr. Darwin's meaning; who, by *a strange inversion of reasoning*, seems to think Absolute Ignorance fully qualified to take the place of Absolute Wisdom in all of the achievements of creative skill. (R. MacKenzie Beverley, quoted in Dennett 2009a, p. 10061, capitalization in original, my italics; see also Dennett 2013b, p. 570, and Dennett 2017, pp. 53–54)

Dennett, however, finds this to be an accurate description of Darwin's theory, and applies it to Turing:

> IN ORDER TO BE A PERFECT AND BEAUTIFUL COMPUTING MACHINE, IT IS NOT REQUISITE TO KNOW WHAT ARITHMETIC IS. (Dennett 2009a, p. 10061; Dennett 2013b, p. 570; Dennett 2017, p. 55)

The "strange inversion" concerns the apparent paradox that "intelligent" behavior can "emerge" from "unintelligent" or "mechanical" behavior:

> From a 'top-down' perspective, human cognition includes complex sequential processes, often involving language or other forms of symbolic representation, as in mathematical calculation. Yet from a 'bottom-up' view, *cognition is nothing but the simple firings of neurons.* Cognitive scientists face the problem of how to reconcile these very different perspectives. (Copeland and Proudfoot, 2017, pp. 313–314)

As an argument against AI's success, the "vitalistic" argument is really nothing more than an appeal to the mystery of cognition. Once the "trick" behind the "magic" of cognition is revealed, either we have to admit that there was nothing magical about it after all, or we have to change our words and our theories about cognition to encompass computation.

Further Reading:
For more on vitalism, especially in connection with Gödel and Turing Machines(!), see Lethen 2020.

Philosophers who hold that cognition is reducible to physics, etc., *and* that psychological language can be *replaced* by physical (etc.) language include Paul and Patricia Churchland (as cited in MacFarquhar 2007, pp. 68–69). Reductionists who *don't* think that such language is eliminable include Oppenheim and Putnam 1958; Fodor 1974.

For a science fiction version of Leibniz's mill, see Asimov 1987.

An alternative spatial metaphor to the "bottom up/top down" view is a view from "the inside" (Leibniz imagined being *inside* the mill) vs. "the outside". The latter metaphor is reminiscent of Turing's so-called "response-dependent" view as proposed by Proudfoot (2017b), as well as Dennett's "intentional stance" view; see the Further Readings at the end of §2.4.1.

4.5 Mathematical Arguments against AI's Success

> Computationalism [the hypothesis that all cognition ... is the execution of some partial recursive functions] would be false if thinking ... involves executing functions that are equivalent to the halting problem.
> —Eric Dietrich (1993, §§2, 6)

To sum up so far, the practical arguments against AI are invalid at worst or can be taken as setting up research projects at best. And the biological arguments are either too parochial (worrying more about how human cognition is *actually* implemented) or too vague (at current states of research). What both such arguments do point to are *possible* limitations to a fully computable theory of cognition. The vitalistic arguments, on the other hand, concern what we are willing to accept as an answer to the question of what cognition is.

However, the only fully convincing argument that cognition is *not* computable needs to be a mathematical or logical one. The great, turn-of-the-20th-century mathematician David Hilbert (1900, p. 445) believed that all mathematical problems had to be solvable "by a finite number of purely logical processes" or else be able to be proved impossible:

> This conviction of the solvability of every mathematical problem is a powerful incentive to the worker. We hear within us the perpetual call: There is the problem. Seek its solution. You can find it by pure reason, for in mathematics there is no *ignorabimus* ["We will not know"].

But then Gödel proved that there were some mathematical sentences that could *not* be proved (see §5.4.1). Can cognition be proved to be uncomputable?

A proof of AI's computational impossibility would not only be the strongest argument against the eventual success of AI, but—given the hypothesis that cognition is computable—it would be the only legitimate argument against it. And the only way to prove that something is not computable is with a mathematical or logical proof, because, after all, computability is a branch of mathematics and logic.

How do you prove that something *is* computable? By providing a computer program that computes it! And how do you know that the program computes *that* function? By comparing its input-output behavior with the inputs and outputs of the function. In the case of cognition, this is where a Turing-like Test comes into the picture.

Now, it is quite possible that cognition is not computable. Not everything is computable. More precisely, there are perfectly good mathematical functions that are not computable. The most famous of these is the Halting Problem, a version of which was proved to be non-computable by Turing in his 1936 paper.

Those who argue that AI *won't* succeed are arguing (or should be arguing!) that cognition is *not* computable. How do you prove that something is *not* computable? Not by failing to find a computer program for it! (Recall the epigraph to §4.) Instead, you typically have to argue that the assumption that the function *is* computable leads to a contradiction.

This is known as an argument by *reductio ad absurdum*—reduction to absurdity—or "proof by contradiction": To prove that $\neg P$ (i.e., that P is not the case, e.g., that cognition is not computable), you *suppose* "for the sake of the argument" (i.e., you *make believe*) that P *is* the case and then show that that assumption leads to a contradiction—an "absurdity". Then, by the valid argument form known as "Modus Tollens"—from $A \rightarrow B$ and $\neg B$, you may infer $\neg A$—you conclude that your assumption (P) was wrong, i.e., that $\neg P$ *is* the case. (For simple examples, see Sidebar E: *Reductio* Arguments.)

The Halting Problem concerns whether it is possible to write a *single* computer program H that behaves as follows: H takes as input *any* computer program C and *any* input i to C, and it outputs either the message 'C halts on i' if C halts on i or else the message 'C goes into an infinite loop on i' if C does not halt on i.

It would be nice to know ahead of time if your program C is going to go into an infinite loop. This would be very useful for introductory computer-programming teachers and students, and for software engineers in general. (Charlesworth 2014, pp. 442–448, argues that many important mathematical theorems and conjectures can be restated as halting problems.) Infinite loops are, generally speaking, Bad Things. After all, one thing you never want in a computer program is an *uninten-*

tional infinite loop. But sometimes you might *want* one: Turing's original algorithms in his 1936 paper were designed to compute the infinite decimal expansion of real numbers; they were not supposed to halt. Moreover, you *don't* want an automated teller machine to halt—you *do* want it to behave in an infinite loop so that it is always ready to accept new input from a new customer. And, presumably, algorithms for cognition had better not halt (at least during the life of the cognitive agent)!

It would, however, be very useful to have a *single* program, H, that could quickly check to see if *any* program that someone writes has an infinite loop in it. But no such program can be written; the function that H would have to implement is not computable! It can be proved that the assumption that it *is* computable leads to a contradiction. "The reason, essentially, is that the scope of the task covers *all* computer programs, **including the program of the very computer attempting the task**" (Copeland, 2017a, p. 61, original italics, my boldface). (For a sketch of a proof, see Sidebar F: The Halting Problem.)

That is the kind of argument that would show that cognition (or some aspect of it) is not computable. But no one has supplied such a proof. Many have tried, but no such attempts have been universally accepted. (An argument that comes close to being of this type will be discussed in §5.4.) That, of course, does not mean that no such proof is possible! But it does mean that there is no (current) reason to believe that cognition is not computable. Hence, there is no (current) reason to believe that AI will not succeed.

And that is why I said, at the beginning, that I take the scientifically optimistic view that there is no (known) logical reason to think that AI won't succeed.

Further Reading:
Determining what an algorithm computes is not as straightforward as it may seem. If the algorithm has a bug, is it a "bad" algorithm for what it was designed to compute, or is it a "good" algorithm for what it actually computes? As Gurevich (1999, p. 98) says, "Errors are in the eye of the beholder." See Suber 1988; Buechner 2011, 2018; and the discussion in Rapaport 2023a, §15.1.1.

Turing-*like* tests don't have to be exactly Turing's; for examples of other such tests, see Marcus et al. 2016, as well as Hernández-Orallo 2020 and other papers in Damassino and Novelli 2020.

4.6 Summary

Will AI fail? We can say that

6. AI will fail catastrophically only if all aspects of cognition can be shown to be non-computable.

But we saw in §3.3 that

4. There are many essential aspects of cognition that have been shown to be computable.

So, we can conclude that

7. AI will not fail catastrophically.

Thus, at worst, AI might fail *non*-catastrophically:

8. AI will fail non-catastrophically only if there is some essential aspect of cognition that can be shown to be non-computable.

But

9. There is no generally accepted argument to that effect . . .

. . . with some possible exceptions that we will explore in the next section.

5 Will *All* of AI Succeed?

5.1 The Cognitive Computability Thesis

The machine sitting at that desk was no longer a man; it was a busy New York broker, moved by buzzing wheels and uncoiling springs.
—O. Henry (1906a, p. 68)

I said at the end of §4.5 that the failure to show mathematically that (some aspect of) cognition was not computable did not imply that no such proof was possible. There is, however, another reason to think that such a proof of its non-computability might, in fact, be impossible.

Computation is a precise mathematical notion. More accurately, there is a precise mathematical notion of computation, namely, Turing Machine computability and its equivalents. But there is also an informal (or, at any rate, less precise) notion of computation, namely, what it is that human "computers" do when they do mathematical calculations. The Church-Turing Computability Thesis states that the informal notion of computation (the kind of thing that humans do) is equivalent to (or entirely captured by) the formal mathematical notion of computation (as embodied in the theory of Turing Machines and their logical equivalents).

Cognition is also a vague or informal notion (Moor 1976, p. 250; Cummins and Schwarz 1991, p. 70). Thus, the working hypothesis of AI—that cognition is computable—is akin to the Computability Thesis in that both equate an informal notion with a formal one. And it is generally agreed that one cannot formally prove such an equivalence:

> A theorem relates mathematical objects; its proof is a mathematical proof. A thesis relates mathematical and non-mathematical objects. It cannot have a mathematical proof, but it may be affirmed or negated by experimentation. It may also have some kind of speculative justification. (Gurevich, 1999, p. 100)

Despite what I just quoted Gurevich as saying, some do believe that the Computability Thesis can be formally proved (e.g., Dershowitz and Gurevich 2008). But they do that by axiomatizing—i.e., formalizing—the informal notion of computation and then showing that that axiomatization is equivalent to, say, Turing Machines. But that just pushes the equivalence thesis back one step: How do you formally prove that the formal axiomatization captures the informal notion?

I argued in §4.5 that to prove that something is not computable, a mathematical argument is needed. But, if cognition is an informal notion, then no mathematical argument for its non-computability can be provided!

Where does this leave us? First, that "cognition" is vague implies that its computability may have to be heuristic, at best. Second, to *support* this "computability thesis" for cognition, we need to continue to do AI research. We need to see which individual aspects of cognition are computable—ones that may, perhaps, be more easily formally characterized: Is language use computable? Is visual perception computable? Is reasoning computable? And so on. And the test for whether such an aspect of cognition is computable is to see if a computer programmed with a computational implementation of it passes a Turing-like Test (or comes heuristically close to passing one).

To *refute* the cognitive computability thesis, we would need to find some essential aspect of cognition that is not computable. And to do that we would need to formalize that aspect of cognition sufficiently so that its computability could be mathematically refuted in the manner of the Halting Problem. But that hasn't yet been done. (Even if it turns out that certain aspects of cognition are too practically difficult to be computationally implemented or are too dependent on their physical implementation to be purely computational, one would still want to know why this is the case.)

Are there any candidates for such aspects? Three stand out: consciousness in general; one particular aspect of consciousness: qualia; and mathematical cognition.

Further Reading:
For attempts to prove the Computability Thesis, see Gandy 1988; Mendelson 1990; Stewart Shapiro 1993, 2013; Folina 1998; Sieg 1994, 2000, 2008a; Kripke 2013; and the discussion in Rapaport 2023a, §§7.4.4, 10.2.

5.2 Is Consciousness Computable?

> . . . there is no obstacle *in principle* to human-level AI intelligence (possibly excepting phenomenal consciousness).
> —Margaret Boden (2018, p. 136)

> As impressive as machine learning is, we have not achieved anything close to artificial general intelligence, a term that refer[s] to something like true comprehension of a topic or to an artificial system that achieves true consciousness or self-awareness. Defining these terms can be tricky, controversial, or even impossible. Personally, Ive never seen a formal definition of consciousness that captures my intuitive notion of the concept. I suspect we will never achieve artificial general intelligence in the strong sense, even if P=NP.
> —Lance Fortnow (2022, p.82, col. 1)

To delve too deeply into the nature of consciousness would take us too far afield. Very briefly, and with a great deal of oversimplification, there are (at least) two different, though related, concepts of consciousness. The philosopher Ned Block has called them "phenomenal" and "access" consciousness:

> Phenomenal consciousness is experience; the phenomenally conscious aspect of a state is what it is like to be in that state. The mark of access-consciousness, by contrast, is availability for use in reasoning and rationally guiding speech and action. (Block, 1995, p. 227)

And the philosopher David J. Chalmers has differentiated them as the "easy" and "hard" problems of consciousness:

> The easy problems of consciousness are those that seem directly susceptible to the standard methods of cognitive science, whereby a phenomenon is explained in terms of computational or neural mechanisms. The hard problems are those that seem to resist those methods. . . . The really hard problem of consciousness is the problem of experience. (Chalmers, 1995, §2)

Access consciousness is the (relatively!) easy problem that, from my scientifically optimistic view, is (likely to be) computable, and there are many computational theories of it. Phenomenal consciousness is the hard problem that has (so far) proved to be difficult to compute. One reason for its difficulty is that it is "private" (or subjective) and "ineffable".

It is *private* in the sense that only the cognitive agent who is subjectively experiencing something (such as a particular color, or a sound, or a pain) *can* experience it. Hence, only the experiencer can know what it is, or what it is like to experience it (Nagel, 1974). An "objective", external observer—such as a scientist or computer programmer attempting to develop a scientific or computational theory of it—cannot experience anyone *else's* experience, so there seems to be no way to know what anyone else's experience is like, or even if they are really having one as opposed to merely *saying* that they are having one.

And phenomenal consciousness is *ineffable*—literally, inexpressible—because mere words cannot convey the exact nature of the experience. At best, they can *describe* the abstract *structure* of the experience. (We'll come back to this in §5.3.)

This is not to say that there have not been attempts to develop computational theories of consciousness (Boden, 2018, Ch. 6). But, as noted, it is going to be difficult to determine whether any will succeed, as we'll see in the next section.

Further Reading:
An excellent history of the various philosophical debates on consciousness is Strawson 2015. See also Strawson 2005, p.47, on AI and conscious experience.

For an analysis of Block's theory, see Kriegel 2006. Boden (2018, p. 108), Sloman (2020b), and Dietrich et al. 2021, pp. 171–196, distinguish many more varieties of consciousness than the two discussed here.

On computational theories of consciousness, see Dehaene et al. 2017, p. 486, col. 3; p. 492, col. 1. Dennett 1992 discusses the "easy" problem in depth. Chalmers 1996 discusses the "hard" problem in depth. For a bibliography of computational theories of consciousness, see https://cse.buffalo.edu/~rapaport/719/csnessrdgs.html. For a recent attempt by two well-respected theorists of computation at formalizing a notion of consciousness (both access and phenomenal) in the way that Turing formalized computation, see Blum and Blum 2021. On the other hand, Arvan and Maley 2022 argue that if "the panpsychist view that phenomenal consciousness ... exist[s] at a microphysical level of reality", then "digital AI ... may lack coherent phenomenally conscious experiences" (pp. 1/22, 17/22).

5.3 Are Qualia Computable?

Are we after all nothing but a sort of robot or machine? or would certain sorts of robots or machine after all be nothing but a sort of us?
—Keith Gunderson (1964, p. 196, fn. 1)

Admittedly, it's highly counter-intuitive to suggest that tin-can computers could really experience blueness or pain, or really understand language. But qualia being caused *by neuroprotein* is no less counter-intuitive, no less philosophically problematic. (So, something that's counter-intuitive may nevertheless be true.)
—Margaret Boden (2018, p. 121)

The red I see in the eye with the new lens is way different than the red I see in my other eye. Which begs the question ... which red is the correct red? And if there's no one around to see it except, say, a dog ... is there any red at all?
—"Funky Winkerbean" comic strip (12 February 2021)[20]

A technical philosophical term for phenomenal experiences is 'qualia' (singular, 'quale'; a Latin term meaning "quality"). One way to make the idea vivid is to think of the color red or of a red object, such as certain fire engines. We can all agree that such a fire engine is red, but is it possible that the private, subjective, visual experience you have when looking at it—your red quale—differs from mine? The answer to *that* question is "yes" (Block, 1999). In fact, the colors that I see with my left eye differ slightly from those that I see with my right eye (at least under certain very sunny lighting conditions). I know this because I can compare those colors (by successively closing and opening each eye), and there are behavioral differences that can be experimentally identified when people's color perceptions differ. (This is how color-blindness is detected.)

The interesting question arises when no comparisons are possible or when there are no behavioral differences. Perhaps my quale that I use the word 'red' for and that I experience when I look at that fire engine is like the quale that you experience when we both look at grass (and that we both call 'green'), and vice versa: Our qualia would be "inverted". Or maybe you have no quale at all when you look at the fire engine, even though you *say* that it is red! Cognitive agents who lack qualia—whose qualia are "absent"—are called 'zombies' by philosophers.

Those who believe that philosophical zombies are possible tend to also believe that qualia are something over and above any "easy problem" or "access" kind of

[20]https://comicskingdom.com/funky-winkerbean/2021-02-12

consciousness and would have to be explained in some way. But the standard computational way of explanation is often identified with a "functional" or "structural" way of explanation in terms solely of inputs, processing, and outputs, and not in terms of any underlying physical embodiment or implementation. Such computational explanations seem not to be able to account for qualia.

Others don't believe that zombies are possible, or they believe that *we* are zombies in the sense that any sufficiently complex organism whose cognition can be explained as part of a solution to the "easy" problem of consciousness would either have to have qualia, or would have to act as if they had them (e.g., Dennett in various essays, as well as in conversation).

If qualia don't exist, then there's no computational problem to be solved. If they do exist, then either they are computable, or else they aren't. One attitude that some have towards the computability of qualia is that they are simply too difficult or puzzling to be able to be understood computationally (see, e.g., Piper 2012). This falls under the heading of a practical argument against AI, as we discussed in §4.1.

If they aren't computable, then it might simply be the case that they are not an essential aspect of cognition that has to be explained computationally but are instead a feature of the physical implementation of the cognitive agent. (I'll give an example in a moment; see also Rapaport 2023b).

In any case, it seems to be logically impossible to know what someone else's qualia are like (or even if they have them). The philosopher Ludwig Wittgenstein argued against the possibility of such a "private language" with his "beetle box" analogy:

> Suppose everyone had a box with something in it: we call it a "beetle". No one can look into anyone else's box, and everyone says he knows what a beetle is only by looking at his beetle. —Here it would be quite possible for everyone to have something different in his box. One might even imagine such a thing constantly changing. —But suppose the word "beetle" had a use in these people's language? —If so it would not be used as the name of a thing. The thing in the box has no place in the language-game at all; not even as a something: for the box might even be empty. —No, one can 'divide through' by the thing in the box; it cancels out, whatever it is. (Wittgenstein, 1958, §293, https://web.stanford.edu/~paulsko/Wittgenstein293.html)

So the inverted and absent qualia problems don't necessarily show that qualia are uncomputable, but perhaps only that they are untestable, in the sense that, even if they are computable, we may never know if they are! (See the discussion in Dietrich et al. 2021, pp. 125–130.)

Others argue that they may be computable (Chalmers, 2011; Sloman and Chrisley, 2003; Dehaene et al., 2018; Blum and Blum, 2021). Here's one way to think about it:[21] Besides colors, pains are another standard example of qualia: Can computers feel pain? Consider the following computational implementation of pain (all of which, by the way, can be done with current technology): Imagine a computer terminal with a pressure-sensitive device hooked up to the central processing unit of the computer in a certain way that I'll specify in a moment. Program the computer with a *very* user-friendly operating system that allows the following sort of interaction (comments in parentheses):

User: (Logs in, as, say "rapaport")

System: Hi there, Bill! How are you? What can I do for you today?

(Assume that this very-user-friendly greeting only occurs at the first login, and that the operating system is capable of some limited, but reasonable, natural-language interaction of the sort discussed in, e.g., Shapiro and Kwasny 1975.)

User: I'd like to finish typing the essay that I was working on yesterday—file "aidebate.tex".

System: No problem; here it is!

(The file is opened. The user edits the file, closes it, and then hits the terminal sharply on the pressure-sensitive device. Assume that this device is wired to the computer in such a way that any sharp blow sends a signal to the central processing unit that causes the operating system to switch from *very*-user-friendly mode to "normal" mode.)

System: File "aidebate.tex" modified and closed. Next command:

User: I'd like to read my mail, please.
(System runs mail program without comment. User exits mail program.)

System: Next command:

(User logs off; logging off in the context of having struck the pressure-sensitive device causes the operating system to switch to yet another mode. The next day, User logs in . . .)

System: Rapaport. Oh yeah; I remember you. You hit me yesterday. That hurt!

[21]The following idea is due to Stuart C. Shapiro (personal communication, ca. late 1980s). See the discussion in Rapaport 2005, §2.3.1.

In this scenario, we have a computer with an AI operating system that seems to be exhibiting pain behavior. Taking into account the differences between the computer and a human, and the limitations of the natural-language interface, it is reasonable to infer (or assume) that the computer was in pain when I hit it. But did it *feel* pain? One can argue that it did, on the grounds that it sensed (at least, it received) the signal sent from the pressure-sensitive device. When a human feels pain, the human senses (that is, the brain receives) a signal sent from the source of the pain. A question for you to think about is whether the reception of that signal would be the computer's pain quale.

Further Reading:
On the "inverted spectrum" (originally due to Locke 1694), see Byrne 2020. On absent qualia, see Tye 2021. For some real-life examples akin to these, see Jarvis 2021.

On philosophical zombies, see Kirk 1974; Chalmers 1996; and other references at http://www.cse.buffalo.edu/~rapaport/719/csnessrdgs.html#zombies. On whether we are zombies, besides Dennett's writings, see also Dehaene et al. 2017, p. 492; Dehaene et al. 2018.

For more detailed attempts at computational implementations of qualia, see Shapiro and Bona 2010, §6.7, p. 321, and Blum and Blum 2021. For contrasting views and other discussions on sensations and qualia, see Dennett 1978; O'Regan 2011, esp. Ch. 8; and Tye 2021.

5.4 Is Mathematical Cognition Computable?

Two kinds of arguments have been proposed to show that aspects of mathematical cognition might not be computable. Obviously, except for provably uncomputable functions, most of mathematics itself is computable; that's not the issue. Rather, the question is whether the cognitive ability to do mathematics—in particular, to discover and prove theorems—is computable.

5.4.1 Gödel's Incompleteness Theorem

The first such argument, originally due to the philosopher John R. Lucas (1961), is based on Gödel's incompleteness theorem, so we'll look at that first. But before we can do that, there are a few preliminary notions:

Syntax has to do with the grammar and proof theory of a formal system—what its (uninterpreted) symbols (or "marks") are, how they are related to each other, and how they can be manipulated. Arguably, this is at the heart of computation, certainly at the heart of the Turing Machine.

Semantics has to do with the relation of the formal system to the (external) world; it is the realm of meaning (more specifically, reference or denotation) and truth.

Provability is a syntactic notion, specifically, which sentences can be logically inferred from which other ones according to (syntactic) rules of inference.

Truth is a semantic notion—having to do with the relation of the system's symbols to something *else* that the system describes, such as a model, or the real world. Provability and truth are two distinct things.

A *consistent* formal system is one that does not contain a sentence S such that both S and $\neg S$ are provable in that system. In a consistent system, all provable sentences are true.

Conversely, a *semantically complete* system is one in which all true sentences are provable.

Gödel's (first) incompleteness theorem states, roughly, that any consistent formal system of first-order logic that includes Peano's axioms for arithmetic (see the Glossary) is *syntactically incomplete* in the sense that there is a sentence S expressible in the system such that neither S nor $\neg S$ is provable in that system.

Because exactly one of those two *unprovable* sentences S and $\neg S$ can be *true*, we know from Gödel's theorem that there is a true statement of the system that cannot be proved in the system (though we don't necessarily know which of the two it is). The true statement might, however, be provable in some other consistent formal system.

Gödel's proof is "constructive", in the sense that it exhibits the unprovable sentence: It constructs it algorithmically. An *informal* version of the proof of Gödel's theorem considers a self-referential sentence like G:

(G) G is unprovable.

Either G is true or else it is false. Suppose that it is false. Then what it says is not true. What it says is that it is unprovable. If that's not true, then G *is* provable. But, first-order logic is "sound", which means that, if its axioms are true and if its inference rules preserve truth, then all provable sentences are true. So G, being provable, is true, contradicting our assumption that it was false. (This is another example of a *reductio* argument.) So, G is *not* false; i.e., it is true. And if it is true, then what it says is true. What G says is that G itself is unprovable. So, G is true but unprovable. (That proof-sketch didn't use first-order logic or arithmetic. But Gödel's actual proof does.)

Further Reading:
Philosophers of computer science argue about whether it is syntax or else semantics that is at the heart of computation; for discussion and further references, see Piccinini 2015, 2020; Rapaport 2017a; Rapaport 2023a, Ch. 16; Rapaport 2020a; and Shagrir 2022. On the relation of syntax to semantics, see Rapaport 2017b.

On first-order logic, see any introductory logic text or Rapaport 1992a. On Peano's axioms, see Peano 1889; Dedekind 1890; Kennedy 1968; Joyce 2005; and https://en.wikipedia.org/wiki/Peano_axioms.
Note to my co-author and reviewers: Do we need a sidebar on first-order logic and Peano's axioms? Or do the Glossary entries suffice?

For popular treatments of Gödel's Theorem, see Hofstadter 1979; Nagel et al. 2001. For more technical, but still accessible, treatments, see Franzén 2005; Raatikainen 2020.

5.4.2 Lucas's Argument

Lucas uses Gödel's theorem to try to show that mathematical cognition is not computable:

> Gödel's theorem must apply to cybernetical machines [i.e., computers], because it is of the essence of being a machine, that it should be a concrete instantiation of a formal system. *It follows that given any machine which is consistent and capable of doing simple arithmetic, there is a formula which it is incapable of producing as being true—i.e., the formula is unprovable-in-the-system—but which we can see to be true.* It follows that no machine can be a complete or adequate model of the mind, that minds are essentially different from machines. (Lucas, 1961, p. 113)

Many people accept some version of this argument; a recent one is due to the mathematical physicist Roger Penrose (1989). But many more do not. A standard objection to Lucas's argument is to deny the italicized sentence in the above quotation. All that follows from Gödel's proof is that, *if* the formal system is consistent, then G is true. But Gödel also proved that the formal system cannot prove its own consistency. Note that, even though a computer programmed with first-order logic plus Peano's axioms can state a true theorem that it cannot prove, a *different* computer could both prove that the first computer's G was unprovable and also "see" that it was true. This simple response probably wouldn't satisfy the fans of Lucas and Penrose; so, the interested reader is urged to read some of the more technical responses.

Turing had made a similar observation ten years earlier:

> [O]ne can show that however the machine [i.e., a computer] is constructed there are bound to be cases where the machine *fails to give an answer* [to a mathematical question], but a mathematician *would be able to*. On the other hand, the machine has certain advantages over the mathematician. Whatever it does can be relied upon, assuming no mechanical 'breakdown', whereas the mathematician makes a certain proportion of mistakes. I believe that this danger of the mathematician making mistakes is an unavoidable corollary of his [sic] power of sometimes hitting upon an entirely new method.
> (Turing, 1951b, p. 256, my italics)

It's not obvious what Turing was alluding to when he said, "there are bound to be cases where the machine fails to give an answer, but a mathematician would be able to." One possibility is that he's referring to Gödel's incompleteness theorem. If a Turing Machine is programmed to prove theorems in Peano arithmetic, then, by Gödel's theorem, there will be a *true* statement of arithmetic that it cannot *prove* to be a *theorem*—that is, to which it "fails to give an answer" *in one sense*. A human mathematician, however, could show *by other means* (but not prove as a theorem in that system!) that the undecidable statement was *true*—that is, the human "would be able to" give an answer to the mathematical question, *in a different sense*.

That is, there are two ways to "give an answer": An answer can be given by syntactically proving a theorem or else by semantically showing a statement to be true.

Turing's comments about the machine's reliability and the human's unreliability may be related to Gödel's requirement that the formal system be consistent. Presumably, human mathematicians aren't, and thus can "hit upon entirely new methods", because—in classical first-order logic—from an inconsistency, *anything* can be proved!

But the important point that I want to make is that such Lucasian arguments are not generally accepted in the way that, say, proofs of the uncomputability of the Halting Problem are. (If they were, we would not be having this debate!) Most logicians think that Lucas and Penrose are wrong.

Further Reading:
The classic reply to Lucas's sort of argument is Putnam 1960, pp. 152–153 (which predates Lucas). For a summary of Putnam's response, see T. McCarthy 2018, pp. 93–95. Other replies to Lucas include Benacerraf 1967 (Lucas 1968 is a response); Boolos 1968; Lewis 1969, 1979; LaForte et al. 1998; Feferman 2009; and Stewart Shapiro 2016. See also Stuart C. Shapiro 1995. For surveys, see, e.g., Franzén 2005, §2.9 and Ch. 6; Raatikainen 2020, §6.3; and https://en.wikipedia.org/wiki/Penrose%E2%80%93Lucas_argument.

For some other remarks on the mathematical abilities of humans vs. machines, see H. Wang 1957, p. 92; and Davis 1990, 1993, who challenges Penrose's 1989 argument that Gödel's Theorem can be viewed as a kind of "hypercomputation" (see Sidebar B.1). On this point, see the discussion in Copeland 2002a, §§1.18–1.18.1); Dennett 1995, Ch. 15 (on Penrose); and Copeland and Shagrir 2013.

On Gödel, Turing, and mathematical ability, see Sieg 2007, Feferman 2011, and the excellent summary of the issues in Koellner 2018. Sieg's essay paints the following picture suggested by some of Gödel's remarks: If individual *brains* can be considered to be Turing Machines, and if individual *minds* are implemented in brains, then the "other means" that are mentioned above might be obtained from individual brains working together (for example, two mathematicians), thus yielding a "larger" "mind". (Goodman 1984 explores this concept in the context of epistemic logic.) That is, each brain—considered as a formal system—can get information from other brains that it couldn't have gotten on its own. Note that this might be modelable as interactive computing (see Sidebar B). Dennett (1995, p. 380) makes a similar observation when he notes that "Science … is not just a matter of making mistakes …"—as the philosopher Karl Popper (1953) suggested when he argued that a field was scientific to the extent that it was falsifiable—"… but of making mistakes in public … in the hopes of getting the others to help with the corrections."

"Relevance" logics are one way of constraining inconsistences so that they don't logically imply every statement; see Anderson and Belnap 1975; Anderson et al. 1992. For applications of relevance logic to AI, see Shapiro and Wand 1976; Martins and Shapiro 1988.

5.4.3 Turing's Argument

The second argument concerning the possible uncomputability of mathematical cognition focuses on a distinction that Turing made between mathematical "ingenuity" and mathematical "intuition":

> Mathematical reasoning may be regarded rather schematically as the exercise of a combination of two faculties, which we may call *intuition* and *ingenuity*. The activity of the intuition consists in making spontaneous judgments which are not the result of conscious trains of reasoning. … The exercise

> of ingenuity in mathematics consists in aiding the intuition through suitable arrangements of propositions, and perhaps geometrical figures or drawings. It is intended that when these are really well arranged validity of the intuitive steps which are required cannot seriously be doubted.
> (Turing, 1939, §11, pp. 214–215)

Turing's comments may be related to Lucas's argument in that perhaps mathematical "ingenuity" is computable but mathematical "intuition" is not (LaForte et al., 1998, §3.2).

More recently, Sloman (2020a,b) has argued that certain kinds of mathematical "intuitions" (to use Turing's term) that humans have used to generate mathematical ideas are not obviously computable in any straightforward way. More specifically, he has argued that

> . . . *digital computers cannot replicate ancient forms of mathematical cognition*. . . .
>
> You may be able to get a feel for what I'm talking about 'from the inside' by answering this question, which you are unlikely to have encountered previously:
>
> > If an object is a convex polyhedron (i.e. it is bounded by plane surfaces intersecting in straight lines forming edges of the surfaces, and where edges meet there are vertices), it is always possible to slice off a single vertex using a planar cut.
>
> Question: what does that do to the number of vertices, edges and faces, and how do you know? . . .
>
> When you have found an answer and a proof that satisfies you think about how easy or difficult it would be to get a computer to go through your process of discovery.
>
> I think biological evolution discovered powerful modes of spatial reasoning used in animal brains long before human mathematicians started using their logical abilities to produce systematic treatments of topology, geometry and arithmetic. (Personal correspondence, 15 February 2020, my italics)

Note that the kinds of spatial reasoning that Sloman mentions echo Turing's 1939 remarks about "geometrical figures".

But recall that, on my view, for AI to "succeed" is for it to come up with computable versions of the cognitive functions that humans (or other cognitive agents) exhibit. It does not have to show that the methods that humans use are themselves computa*tional*, only that they are comput*able*. That is, it does not have to show that the (human) brain *is* a computer, only that what the brain *does* is computable. So it does not have to show that a computer can "replicate" certain "forms of mathematical cognition". It only has to show that the results of that cognition (of that "intuition") can be arrived at by computational means (by "ingenuity").

Further Reading:
Turing's ingenuity-intuition distinction is paralleled by his "discipline-initiative" distinction (Turing, 1948):

> If the untrained infant's mind is to become an intelligent one, it must acquire both discipline and initiative. So far we have been considering only discipline [via the universal machine]. ... But discipline is certainly not enough in itself to produce intelligence. That which is required in addition we call initiative. This statement will have to serve as a definition. Our task is to discover the nature of this residue as it occurs in man [sic], *and to try and copy it in machines*. (Turing 1948, p. 21 of the online version, my italics; bracketed interpolation about the universal machine from Sieg 2020, p. 193)

On the role of learning, see also the discussion in Sieg 2020, esp. p. 192.

For a related argument about intuition vs. ingenuity, see LaForte et al. 1998, §3.4. Duwell 2021, §3.1, points out that "One can reasonably expect that human capabilities extend beyond what Turing machines are capable of because we do have insight, intuition, or ingenuity."

5.5 Summary: But *X* Is Not Computable!

I have canvassed three phenomena that are said to be features of cognition that are said to be not computable. When faced with an argument about some cognitive feature *X* that is said to be uncomputable, the first question you should consider is whether it really is a(n essential) feature of cognition. Clearly, if it is, then AI researchers need to figure out how to compute it (as B.C. Smith (2019) argues). But it may turn out that *X* might either not be cognitive after all, or might not be essential to cognition. For example, even though qualia may seem to be "psychological", if it turns out that they are a physical *implementation* feature rather than an essential feature of cognition, then it may not matter if they are uncomputable.

We can now elaborate on two of our previous premises:

8. AI will fail non-catastrophically only if there is some essential aspect of cognition that can be shown to be non-computable *or non-heuristically computable.*

9. There is no generally accepted argument to that effect, *with the possible exception of phenomenal consciousness (qualia).*

Further Reading:
The essayist David Brooks (2023) has suggested some "distinctly human skills" that would differentiate a human from a program like ChatGPT that seems to pass a Turing-like test: "A distinct personal voice. … Presentation skills. … A childlike talent for creativity. … Unusual worldviews. … Empathy. … Situational Awareness." Of course, he offers no proof that they are non-computable, and so I think that they are best taken as challenges for AI researchers.

6 Concluding Remarks

> It is customary … to offer a grain of comfort, in the form of a statement that some particularly human characteristic could never be imitated by a machine. … I cannot offer any such comfort, for I believe that no such bounds can be set.
> —Alan Turing (1951a, p. 486)

Cognition has by no means been shown to be fully computable. Indeed, there may be some very computationally recalcitrant aspects of cognition. These might then be declared *not* to be aspects of cognition after all or to be side effects of the implementation. The mathematician and cyberneticist Norbert Wiener said (in a way reminiscent of Lucas's objection):

> … for a long time at least there will always be some level at which the brain is better than the constructed machine, even though this level may shift upwards and upwards. (Wiener, 1960, p. 1357)

This suggests that there are three ways in which AI as computational cognition might succeed:

1. If, as suggested in §5, cognition is a vague concept, then AI's "success" might best be measured in terms of how many aspects of cognition can be shown to be computable. Computable cognition might then asymptotically approach (human) cognitive abilities even if it never fully reaches that limit. Here is where Turing's prediction about the use of words and general educated opinion might be relevant.

2. AI might, of course, completely succeed in the sense of reaching that limit. Turing certainly thought so (as expressed in the epigraph above). Or there might be a *range* of cognitive abilities such that both humans and computers might lie in that range, even if they don't exactly "match". Intelligence of the non-human animal kind may be quite "general" for the animal, even

though not reaching the level of human intelligence. Similarly, human-level AI (HLAI) may not be the same thing as artificial general intelligence (AGI):

> Non-human animals . . . may have an even smaller repertoire of capabilities, but are nonetheless general in that subset. This means that humans can do things animals cannot and AGI will be able to do something no human can. If an AGI is restricted only to domains and capacity of human expertise, it is the same as HLAI. (Yampolskiy, 2020, §3)

3. Or AI might succeed beyond our wildest imaginations, so to speak, and become *more* "intelligent" than humans. This is not beyond the imagination of some philosophers and scientists who believe that such a "singularity" will be exceeded in the not too distant future.

But *no one has yet devised a mathematically acceptable proof that cognition is **not** computable.* To paraphrase an argument in favor of the unity of science due to Oppenheim and Putnam (1958, p. 8), my position is this:[22]

> Can [AI] be attained? [I] certainly do not wish to maintain that it has been *established* that this is the case. But it does not follow, as some philosophers seem to think, that a tentative acceptance of the hypothesis that [AI] can be attained is therefore a mere "act of faith." [I] believe that this hypothesis is *credible*; and [I] shall attempt to support this . . . by providing . . . reasons in its support. [I] therefore think the assumption that [AI] can be attained recommends itself as a *working hypothesis*. That is, [I] believe that it is in accord with the standards of reasonable scientific judgment to tentatively accept this hypothesis and to work on the assumption that further progress can be made in this direction, without claiming that its truth has been established, or denying that success may finally elude us.

And so I conclude that

10. There is no scientific reason to believe that cognition cannot eventually be shown to be computable.

[22]My bracketed '[AI]' replaces their 'unitary science', and my '[I]' replaces their 'we'.

Further Reading:
For overviews of "The Singularity", see Bringsjord and Govindarajulu 2020, §9; https://en.wikipedia.org/wiki/Technological_singularity; and http://www.singularity.com/. As with so much else in computer science and AI, the idea behind the Singularity can be traced back to Turing (1951a, pp. 485–486). For discussions, see Chalmers 2010, esp. §4, "Obstacles to the Singularity"; the anthology edited by Eden et al. (2012); and Davis 2015. A symposium on Chalmers's paper is in *Journal of Consciousness Studies*, Vol. 19, issues 1–2 and 7–8 (the tables of contents are at https://www.ingentaconnect.com/content/imp/jcs), with a reply by Chalmers (2012). Bundy 2017 argues that "Worrying about machines that are too smart distracts us from the real and present threat from machines that are too dumb."

7 Final Summary

Here, then, in outline form is the argument in support of my position in this debate:

1. AI is the branch of computer science that investigates computational theories of cognition.
2. AI's success can be measured in terms of how much of cognition can be shown to be computable.
3. AI will completely succeed if it can show that all (or all essential, or nearly all) aspects of cognition are computable (or heuristically computable).
4. There are many essential aspects of cognition that have been shown to be computable.
5. $\therefore$ AI has partially succeeded.
6. AI will fail catastrophically only if all aspects of cognition can be shown to be non-computable.
7. $\therefore$ AI will not fail catastrophically.
8. AI will fail non-catastrophically only if there is some essential aspect of cognition that can be shown to be non-computable (or non-heuristically computable).
9. There is no generally accepted argument to that effect, with the possible exception of phenomenal consciousness (qualia).
10. $\therefore$ There is no scientific reason to believe that cognition cannot eventually be shown to be computable.
11. $\therefore$ **There is no reason to believe that AI will not succeed.**

I will close with a comment that can be taken as a gloss on Nakra's definition of computer science (from §2.4.1):

> The point of science is not the holy grail but the quest—the searching and the asking.
> —James Gleick (2021, p. 36)

Acknowledgments

Thanks to my colleagues Stuart C. Shapiro and Barry Smith for comments on an earlier version.

A Sidebar: The Chinese Room Argument

If a Martian could learn to speak a human language, or a robot be devised to behave in just the ways that are essential to a language-speaker, an implicit knowledge of the correct theory of meaning for the language could be attributed to the Martian or the robot with as much right as to a human speaker, even though their internal mechanisms were entirely different.
—Michael Dummett (1976, p. 70)

[R]esearchers ... tracked ... unresponsive patients ..., taking EEG recordings .... During each EEG recording, the researchers gave the patients instructions through headphones. ... "Somewhat to our surprise, we found that about 15 percent of patients who were not responding at all had ... brain activation in response to the commands," said Dr. Jan Claassen .... "It suggests that there's some remnant of consciousness there. *However, we don't know if the patients really understood what we were saying. We only know the brain reacted.*"
—Benedict Carey (2019, my italics)

Thirty years after Turing's publication of the Turing Test, Searle published his Chinese Room Argument (CRA) (Searle, 1980, 1982, 1984). In this thought experiment, a human who knows no Chinese is placed in a windowless room (the "Chinese room") along with paper, pencils, and a book containing an English-language algorithm for manipulating certain marks or symbols that are meaningless to the person in the room. Through a slot in one wall of the room, a native speaker of Chinese inputs pieces of paper containing a text written in Chinese along with reading-comprehension questions about that text, also in Chinese. The person in the room—from whose point of view the papers contain nothing but meaningless marks—consults the book and follows its instructions, which describe how to manipulate the marks in certain ways, to write certain marks down on a clean piece of paper, and to output those "responses" through the slot. The native speaker who reads them determines that all of the questions have been answered correctly in Chinese, demonstrating a fluent understanding of Chinese. This is because the rule book of instructions is a complete natural-language-understanding algorithm for Chinese. But, by hypothesis, the person in the room does not understand Chinese. We seem to have a contradiction.

The CRA can be understood as a counterexample to the Turing Test, concluding from this thought experiment that it is possible to pass a Turing Test, yet not really think. The setup of the CRA is identical to the simplified, two-player version of the Turing test: The interrogator is the native Chinese speaker, who has to decide whether the entity being communicated with understands Chinese. The interrogator determines that the entity *does* understand Chinese. This is analogous to deciding that the entity in the simplified Turing Test is a human, rather than a computer. But, according to the hypothesis of the CRA, the entity in fact does *not* understand Chinese. This is analogous to the entity in the simplified Turing Test being a computer. So, the person in the Chinese Room passes the Turing Test without being able to "really" understand; hence, the Turing Test fails.

Or does it? One standard objection to the CRA is what Searle called the "Systems Reply", which points out that the person in the room is not alone. It is the *system* consisting of the person *plus* the instruction book that understands Chinese. After all, the CPU of a computer doesn't add all by itself: It is the system consisting of the CPU *plus* a computer program that adds. This is made explicit in an earlier version of this kind of set-up:[23]

> Consider a box B inside of which we have a man L with a desk, pencils and paper. On one side B has two slots, marked *input* and *output*. If we write a number on paper and pass it through the input slot, L takes it and begins performing certain computations. If and when he finishes, he writes down a number obtained from the computation and passes it back to us through the output slot. Assume further that L has with him explicit deterministic instructions of finite length as to how the computation is to be done. We refer to these instructions as P. Finally, assume that the supply of paper is inexhaustible, and that B can be enlarged in size so that an arbitrarily large amount of paper work can be stored in it in the course of any single computation. ... I think we had better assume, too, that L himself is inexhaustible, since we do not care how long it takes for an output to appear, provided that it does eventually appear after a finite amount of computation. We refer to the system B-L-P as M. ... In the approach of Turing, the symbolism and specifications are such that the entire B-L-P system can be viewed as a digital computer .... Roughly, to use modern computing terms, L becomes the logical component of the computer, and P becomes its program. In Turing's approach, the entire system M is hence called a *Turing machine*. (Rogers, 1959, pp. 115, 117)

Would Searle argue that M (or L?) is not computing?[24]

[23]Arguably, an even earlier version was in a 1954 episode of *I Love Lucy* (Rapaport, 2023a, §18.8.4).

[24]There are many other responses to the CRA: For overviews, see Rey 1986; Hauser 2001; Cole 2020. Rapaport 1986a is an overview written for a semi-popular computer magazine, and Rapaport 1988b is a critical review of Searle 1984. Preston and Bishop 2002 is an anthology of responses to Searle (reviewed in Rapaport 2006b). In 2014, an online series of articles providing background for

B Sidebar: Computability

[C]omputer science is not really that much about computers. What computer science is mostly about is *computation*, a certain kind of process such as sorting a list of numbers, compressing an audio file, or removing red-eye from a digital picture. The process is typically carried out by an electronic computer of course, but it might also be carried out by a person or by a mechanical device of some sort.

The hypothesis underlying AI … is that ordinary *thinking* … is also a computational process, and one that can be studied without too much regard for who or what is doing the thinking.

—Hector J. Levesque (2017, pp. ix–x)

B.1 Varieties of Computability

Four notions of (digital) “computability” are relevant to our discussion: an informal one, a formal one, a slightly more general formal one, and an arguably hypothetical one.[25]

The *informal notion* of computation is the one that, from at least 1613 until at least the 1940s only humans did, as in the *New York Times* ad mentioned in §3 and in Turing’s 1936 paper.[26]

There are several *formal notions*: Turing Machines, general recursive functions, the lambda calculus, etc.[27] Importantly, all of these formal versions are logically equivalent, which is one of the primary reasons why it is believed that they completely capture the informal notion. This is codified as the Church-Turing Computability Thesis.[28]

the movie *The Imitation Game* contained two critiques of the CRA: Cole 2014; Horst 2014.

[25]Piccinini 2015 suggests several others, including digital, analog, neural, and “generic”. This sidebar is only concerned with the digital variety.

[26]The year 1613 is the earliest citation for ‘computer’ in the *Oxford English Dictionary*, http://www.oed.com/view/Entry/37975, quoted in Rapaport 2018b, p. 388. The use of ‘computer’ to refer to humans extended beyond the 1940s (even into the 1960s), as told in the 2016 film *Hidden Figures* (https://en.wikipedia.org/wiki/Hidden_Figures); see also Bolden 2016; Natarajan 2017.

[27]On recursive functions, see Sidebar D: Recursion, and Rapaport 2023a, Chs. 7–8, or any textbook on the theory of computation. On the lambda calculus, see “PolR” 2009, §“Alonzo Church’s Lambda-Calculus”; and Alama and Korbmacher 2021. Other formalisms include Post production systems (see Soare 2009, §5.2, p. 380), Markov algorithms (https://en.wikipedia.org/wiki/Markov_algorithm), and register machines (Shepherdson and Sturgis, 1963), as well as any computer programming language that includes facilities for sequence, selection, and repetition; see §B.2.1.

[28]Alonzo Church’s thesis (Church, 1936) was first expressed in terms of his lambda calculus and later in terms of recursive functions (see Soare 2009, p. 372 and §11.1, p. 389). Turing’s thesis (Turing, 1936) was expressed in terms of his *a*-machines (i.e., Turing Machines). Because all of these formalisms are logically equivalent, Robert Soare (2009) has advocated replacing the names

The *slightly more general formal notion* allows for a Turing Machine to interact with users, with other computers, or with the "external" world in general (i.e., the world or the user independent of the computer) via inputs from the external world such as data or sensory input, and outputs to it (including physical manipulation of things in the external world). This is the basis of modern computers. Some computer scientists have argued that this goes significantly—not just "slightly"—"beyond" Turing Machine computation (Wegner, 1997). But other computer scientists (Davis, 2004; Soare, 2013) have observed that this is a standard part of computability theory, due in fact to Turing (1939).[29]

The *hypothetical notion* goes under the name 'hypercomputation' and sometimes includes the interactive computation just mentioned, "trial-and-error" machines (see an example in Sidebar F.1: The Halting Function), and some less physically plausible ones, such as "Zeus" machines that can solve the Halting Problem by executing an infinite number of computer programs in a finite amount of time by "infinite acceleration": performing each step in half the time discussed of the previous step. I will not consider these here, but the literature is well worth reading.[30]

'Church's Thesis', 'Turing's Thesis', and 'the Church-Turing Thesis' with—more simply and more neutrally—'the Computability Thesis'. For an overview of these and other Computability Theses, see Copeland and Shagrir 2019.

[29]For a discussion of the debate over interactive computing, see Rapaport 2023a, §11.8.

[30]For an overview, see Copeland 2002b, 2003. For objections, see Davis 2004, 2006. For discussion, see Rapaport 2023a, Ch. 11. On trial-and-error machines, see Putnam 1965; Gold 1967; Hintikka and Mutanen 1997; Kugel 2002; Soare 2009. On their relationship to Lucas's argument (§5.4.2, above), see T. McCarthy 2018. On Zeus machines, see Boolos and Jeffrey 1974. Perhaps hypercomputation is what P. Wang 2020, p. 83, has in mind when he disputes my characterization of AI as a branch of computer science by making the otherwise puzzling claim that "AI problems are conceptually beyond the scope of computability theory, though AI systems are still implementable in computers."

B.2 Turing Machine Computability

[A] human calculator, provided with pencil and paper and explicit instructions, can be regarded as a kind of Turing machine.
—Alonzo Church (1937)

A man provided with paper, pencil, and rubber [eraser], and subject to strict discipline, is in effect a universal machine.
—Alan Turing (1948, p. 416)

In fact we have been universal computers ever since the age we could follow instructions.
—Chris Bernhardt (2016, p. 12; see also p. 94)

B.2.1 Five Great Insights of Computer Science

The kind of computability that is central to AI is Turing Machine computability and its equivalents (along with its extension to interactive computing).[31] One way to understand this is via what I consider to be the five great insights of computer science.[32] Briefly, these are:

1. The *representational* insight:
   Only 2 nouns are needed to represent information
   (‘0’, ‘1’)

2. The *processsing* insight:
   Only 3 verbs are needed to process information.
   (*move*(left or right), *print*(‘0’ or ‘1’), *erase*)

3. The *structural* insight:
   Only 3 grammar rules are needed to combine actions.
   (sequence, selection (if-then-else), and repetition (while loops))

4. The “*closure*” insight:
   Nothing else is needed.
   (This is the import of the Computability Thesis.)

[31]Kugel 2002 argues that trial-and-error machines are central to AI. For discussion, see Rapaport 2023a, §11.10.2.

[32]For more details, see Rapaport 2023a, §7.4. The exact number of nouns, verbs, or grammar rules depends on the formalism. E.g., some presentations add ‘halt’, ‘read’, or ‘exit’ as verbs, or use recursion as the single rule of grammar, etc. The point is that there is a very minimal set and that nothing else is *needed*. Of course, more nouns, verbs, or grammar rules allow for greater ease or practicality of expression.

5. The *implementation* insight:
   The first three insights can be physically implemented.

B.2.2 Turing Machines

A Turing Machine can be thought of as an abstract, mathematical model of a computer. It consists of two principal parts:

1. an arbitrarily long, paper tape divided into squares (like a roll of toilet paper, except that you never run out (Weizenbaum, 1976)); each square can have either a '0' or a '1' printed on it;[33]

2. a reading and writing device that can move to any square, see what symbol is on it, and then print (overwrite) a new symbol (or erase one) on it.

The "program" for the Turing Machine tells it which square to start looking at on the tape and, on the basis of what is printed on that starting square, to do one or more of the following things:

- move left 1 square
- move right 1 square
- print 0 at the current square
- print 1 at the current square (or perhaps erase the current square).

By the Computability Thesis, every informal computation (every algorithm) can be expressed in this minimal language for a Turing Machine. Even artificial-neural-network and machine-learning algorithms can be written in this language.

It follows that, if cognition is computable, then every aspect of cognition can be expressed in this language. But no one would seriously consider trying to do that! (To do so would be to get mired in "the Turing tar-pit in which everything is possible but nothing of interest is easy" (Perlis, 1982, §54).)

Turing modeled his machine (he called it, simply, an '*a*-machine'—'*a*' for 'automatic') on how humans compute, and his description (in Turing 1936, §9) is well worth reading.[34] Because his theory of *automatic* or machine computation is based on his observation of how *humans* compute, it can really be considered to be the first AI program (Rapaport, 2023a, pp. 74, 158, 174).[35]

[33]Some versions allow only one symbol, together with allowing some squares to be blank. In fact, there can be—and in real computers there are—many different symbols.

[34]For detailed tours through Turing's paper, see Petzold 2008 and Rapaport 2023a, Ch. 8.

[35]See also the discussion of Turing's role in the history of AI in Copeland 2023, esp. p. 20.

B.2.3 Universal Turing Machines

Turing Machines, however, are not programmable, even though they have programs! This is because a given Turing Machine is *defined* by its program. If you change the program, you have a different Turing Machine. So, a Turing Machine can only do one thing: the task that it is programmed for.

However, *Universal* Turing Machines can do many things (as Turing proved in his 1936 paper). A Universal Turing Machine is a Turing Machine. As such, it is programmed to do only one thing: to follow and execute the instructions of a program that is encoded on its tape. Because that program *is* changeable, Universal Turing Machines *are* programmable. Your laptop and your smartphone are physical implementations of a Universal Turing Machine, and their "apps" are the programs that they can execute. Downloading and installing a new app is the physical implementation of encoding a new program on a Universal Turing Machine's tape.

B.3 Efficient Computability

The theoretical efficiency of a computation can be measured in terms of its running time, calculated as a function of the size of its input.

An algorithm is said to be "in *P*" if it is computable in *P*olynomial time, i.e., if the time that it takes for the computation can be expressed as a polynomial of the form $a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$, where x is the size of the input to the algorithm. Such algorithms are considered to be computationally "efficient" or "tractable".

An algorithm is said to be "in *NP*" if it is not in *P*. Such algorithms are said to be computable in *N*on-deterministic *P*olynomial time; i.e., they are computable in *polynomial* time by a *non*-deterministic Turing Machine (which is a Turing Machine that allows multiple "next steps" for a given action, all of which are simultaneously taken). Alternatively, an algorithm, expressed as a decision problem, is in *NP* if its "yes" answer is *verifiable* in *polynomial* time.

An algorithm is said to be "*NP*-complete" if it is in *NP* and if all other *NP* problems can be shown in polynomial time to be equivalent to it. *NP*-complete problems are considered to be the most intractable of all computable problems.

One of the (if not *the only*) major open question in computability theory (more precisely, the branch known as "complexity" theory) is whether $P = NP$. If $P = NP$, then

> We get a beautiful world where everything is easy to compute. We can quickly learn just about everything, and the great mysteries of the world fall quickly, from cures [for] deadly diseases to the nature of the universe.

> The beautiful world also has a dark underbelly, including the loss of privacy and jobs, as there is very little computers cannot figure out or accomplish. (Fortnow, 2013, p. 9)

Most computer scientists believe that $P \neq NP$, but no proof or refutation of that has yet been found. (See Fortnow 2013 for a non-technical discussion.)

C Sidebar: Simulations

> ... the marvellous pulp and fibre of a brain had been substituted by brass and iron, he [Babbage][36] had taught wheelwork *to think, or at least to do the office of thought*.
> —Harry Wilmot Buxton, quoted in Swade 2017, p. 255; my italics

> Turing's conclusion ... is that a *perfect* simulation of a thing serves very nicely for that thing. Big *can* grow out of sufficient, collected littles.
> —Richard Powers (1988, p. 221)

The word 'simulation' sometimes connotes an "imitation" or something "unreal":[37] A simulation of a hurricane is not a real hurricane, because it doesn't get people wet. A simulation of digestion is not real digestion, because it doesn't digest food. And a computer that simulates some process P is not *necessarily* "really" doing P.

But sometimes there is little difference between the real thing and what a computer does: Although a computer simulation (or even a replication) of the daily operations of a bank is not thereby the daily operations of a (real) bank, I *can* do my banking online. In fact, real tellers in real banks perform most of their tasks on a computer. Simulations *can* be used as if they were real, as long as the simulations have causal impact on the real world. In fact, sometimes things like *simulated* hurricanes *can* do something analogous to getting *real* people wet: Children "can have real emotional reactions to entirely imaginary scenarios" (Gopnik, 2009, p. 31); so, of course, can adults, as anyone who has wept at the movies can testify (cf. Schneider 2009).

More importantly, although computer simulations of hurricanes don't get real people wet, they *could* get *simulated* people *simulatedly* wet:

> [A] simulated hurricane would feel very real, and indeed can prove fatal, *to a person who happens to reside in the same simulation*. (Edelman, 2011, p. 3, fn. 3; my italics)[38]

The "person", of course, would have to be a *simulated* person. Similarly,

> Paul Harris (2000) found that even two-year-olds will tell you that if an imaginary teddy [bear] is drinking imaginary tea, then if he spills it the imaginary floor will require imaginary mopping-up. (Gopnik, 2009, p. 29)

[36]Charles Babbage developed an ancestor of the digital computer in the 19th century. For more on Babbage, see http://www-gap.dcs.something-and.ac.uk/~history/Mathematicians/Babbage.html and Rapaport 2023a, §6.4.3.

[37]This sidebar is adapted from Rapaport 2012, pp. 54–55.

[38]See also Dennett 1978; Hofstadter 1981; Rapaport 1986c, p. 274; Shapiro and Rapaport 1991, §(10.)7, p. 252; Rapaport 2005, §3.

This is a matter of the "scope" of the simulation: Are people within the scope of the hurricane simulation or not? If they are not, then the simulation won't get them wet. If they are—i.e., if they are simulated, too—then it will.

But there are cases where a simulation or imitation *is* the real thing. For example:

- a scale model of a scale model of the Statue of Liberty *is* a scale model of the Statue of Liberty,
- a Xerox copy of a document *is* that document, at least for purposes of reading it and even for some legal purposes, and
- a PDF version of a document *is* that document.

More generally, a computer that simulates an *informational process* really is thereby doing that informational process, because a computer simulation of information *is* information:

- A computer simulation of language *is* language:

  > [L]anguage is fundamentally computational. Computational linguistics has a more intimate relationship with computers than many other disciplines that use computers as a tool. When a computational biologist simulates population dynamics, no animals die. When a meteorologist simulates the weather, nothing gets wet. But *computers really do parse sentences. Natural language question-answering systems really do answer questions.* The actions of language are in the same space as computational activities, or alternatively, these particular computational activities are in the same space as language and communication. (Woods, 2010, p. 605, my italics)

- A computer simulation of a picture *is* a picture, hence the success of digital "photography".
- A computer simulation of mathematics *is* mathematics:

  > A simulation of a computation and the computation itself are equivalent: try to simulate the addition of 2 and 3, and the result will be just as good as if you "actually" carried out the addition—that is the nature of numbers. (Edelman, 2008, p. 81)

- A computer simulation of reasoning *is* reasoning. This was surely the claim of Newell, Shaw, and Simon's (1958) Logic Theorist (mentioned in §2.3.1).

And, in general, a computer simulation of cognition *is* cognition. To continue the just-cited quotation from Edelman (2008, p. 81):

> Therefore, if the mind is a computational entity, a simulation of the relevant computations would constitute its fully functional replica.

Or, as Simon put it:

> [A] computer simulation of thinking thinks. It takes problems as its inputs and (sometimes) produces solutions as its outputs. It represents these problems and solutions as symbolic structures, as the human mind does, and performs transformations on them, as the human mind does. ... The materials of thought are symbols—patterns, which can be replicated in a great variety of materials (including neurons and chips), thereby enabling physical symbol systems fashioned of these materials to think. (Simon, 1995, p. 24)

(For more on simulations vs. the real thing, see Sidebar A: The Chinese Room Argument. For more on AI simulations, see Durán 2020.)

D Sidebar: Recursion

Recursive mathematical functions are one of the many logically equivalent formalisms of computation. It is beyond our scope to present a mathematical treatment of them here. (But see any textbook on the theory of computation or Rapaport 2023a, §7.7, for presentations.) But recursion and recursive definitions play such a large role in computer science that it is worth spending some time with the notion.[39]

A recursive definition of some concept *C* consists of two parts. The first part, called the "base case", gives you some explicit examples of *C*. These are not just any old examples, but are considered to be the simplest, or most basic (or "atomic"), instances of *C*—the building blocks from which all other, more complex ("molecular") instances of *C* can be constructed.

The second part of a recursive definition of *C* consists of rules (algorithms, if you will!) that tell you how to construct those more complex instances of *C*. But these rules don't just tell you how to construct the complex instances from just the base cases. Rather, they tell you how to construct the complex instances of *C from any instances of C that have already been constructed* (or have been given as base cases). The first complex instances, of course, will be constructed directly from the base cases. But others, even more complex, will be constructed from the ones that were constructed directly from the base cases, and so on. So, the base cases of a recursive definition tell you how to *begin*. And the recursive cases tell you how to *continue*.

What makes such a definition "recursive" is that simpler instances of *C* "recur" in the definitions of more complex instances. Thus, recursive definitions sometimes *seem* to be circular: After all, we seem to be defining instances of *C* in terms of instances of *C*! But really we are defining "new" (more complex) instances of *C* in terms of *other*—"older" (that is, already constructed), hence simpler—instances of *C*, which is not circular at all. It would only be circular if the *base cases* were somehow defined in terms of themselves. But they are not "defined"; they are given, by fiat.

For example, the structural insight from §B.2 is a recursive definition of the notion of an "instruction": The base cases of the recursion are the primitive verbs of the Turing Machine ('move(*location*)' and 'print(*symbol*)'), and the recursive cases are given by sequence, selection, and repetition.

In recursive function theory, the base cases might be the successor function, the predecessor function, and the projection function, and the recursive cases would be composition (which corresponds to sequencing), conditional definition (which

[39]This sidebar is adapted from Rapaport 2023a, §7.6.5.

corresponds to selection), primitive recursion (which computes the value of a function in terms of its previous value; it corresponds to count-loops), while-recursion (which corresponds to while-loops), and the "mu-operator" (which corresponds to unbounded search). (Again, for details, see, e.g., Rapaport 2023a, §7.7.)

Recursion is considered to be the core of the concept of *computation*. It has been argued that it is also the core of the concept of *language* in particular, and of cognition more generally:

> [The] faculty of language ... in the narrow sense ... only includes recursion and is the only uniquely human component of the faculty of language. (Hauser et al., 2002)

> ... the work of Alonzo Church, Kurt Gödel, Emil Post, and Alan Turing, who established the general theory of computability ... demonstrated how a finite object like the brain could generate an infinite variety of expressions. (Chomsky, 2017)

However, these are highly controversial claims.[40]

[40]To follow the debate, see Pinker and Jackendoff 2005; Fitch et al. 2005; Jackendoff and Pinker 2005. For another take on this debate, see Corballis 2007.

E Sidebar: *Reductio* Arguments

> When one loves one's Art no service seems too hard. That is our premise. This story shall draw a conclusion from it, and show at the same time that the premise is incorrect. That will be a new thing in logic, and a feat in story-telling somewhat older than the great wall of China.
> —O. Henry (1906b, p. 20)

"Modus Tollens" (MT) is the argument form:

> From $A \to B$
> and $\neg B$
> you may infer $\neg A$

The strategy of a proof by contradiction, or *reductio ad absurdum*, is:

```
To prove that $P$, do:
    assume for the sake of argument (i.e., make believe) that $\neg P$;
    derive a contradiction $C$ [Often, $C = (P \& \neg P)$]
    $\therefore \neg P \to C$.
    But $\neg C$ (because $C$ is a contradiction).
    $\therefore \neg\neg P$ (by MT, taking $A = \neg P$ & $B = C$)
Conclude that $P$ (by Double Negation)
```

Here are two famous examples of *reductio* arguments.

$\sqrt{2}$ Is Irrational

1. Suppose, by way of contradiction, that $\sqrt{2}$ *is* rational
(i.e., that it is *not ir*rational).
2. $\therefore$ There are integers a, b such that $\sqrt{2} = a/b$ and such that that fraction a/b is in lowest terms (i.e., that a and b have no common factor).
[Follows from premise 1 by the definition of 'rational'.]
3. Because $\sqrt{2} = a/b$, it follows that $2 = a^2/b^2$.
[By squaring both sides of the equation.]
4. $\therefore$ $2b^2 = a^2$. [By elementary algebra.]
5. $\therefore$ a^2 is even. [Because it is a multiple of 2.]
6. $\therefore$ a is even.
[If a were odd, then a^2 would be odd (by elementary algebra).
$\therefore$ By Modus Tollens, if a^2 is even, then a is even.]
7. $\therefore$ There is some integer k such that $a = 2k$ [By the definition of 'even'.]
8. $\therefore$ $2b^2 = (2k)^2 = 4k^2$. [By elementary algebra.]

9. $\therefore\ b^2 = 2k^2$. [By elementary algebra.]
10. $\therefore b^2$ is even. [By the definition of ‘even’.]
11. $\therefore b$ is even. [As before.]
12. But if both *a* and *b* are even, then they have a common factor (namely, 2), which contradicts the claim above that they had *no* common factor.
13. $\therefore$ Our initial supposition, that $\sqrt{2}$ was rational, was false.
14. $\therefore\ \sqrt{2}$ is irrational.

The Mutilated Chessboard

Consider the following problem:

> An ordinary chessboard has had two squares—one at each end of a diagonal—removed. There is on hand a supply of 31 dominoes, each of whihc is large enough to cover exactly two adjacent squares of the board. Is it possible to lay the dominoes on the mutilated chessboard in such a manner as to cover it completely? (Black, 1952, pp. 157, 433)

Typical chessboards have 64 alternating red and black squares. So, each domino would have to cover one red and one black square. Suppose (by way of contradiction) that it *is* possible to cover the mutilated chessboard with the 31 dominoes. But the two squares that were removed were the same color (look at a chessboard to see why this is the case), let’s say red. Therefore, the mutilated chessboard has 32 black squares, but only 30 red ones. So, it is impossible for the dominos to cover an equal number of them.

(For more on the mutilated chessboard, in the context of AI, see Simon 1995, pp. 33–34.)

F Sidebar: The Halting Problem

You can build an organ which can do anything that can be done, but you cannot build an organ which tells you whether it can be done.
—John von Neumann (1966), cited in Dyson 2012.

F.1 The Halting Function H

The Halting Problem provides an example of a *non-computable* function, that is, a function that cannot be computed using any of the mechanisms of Turing Machines.[41] Recall that a function *is* computable if and only if there is an algorithm (that is, a computer program) that computes it. So, the Halting Problem asks whether there is an *algorithm* (for example, a program for a Turing Machine)—call it the "Halting Algorithm", A_H—that computes the following *function* $H(C,i)$ (call it the "Halting Function"):

$H(C,i)$ takes as input *both*:

1. any algorithm (or computer program) C
   (which we can suppose takes an integer as input),

   and

2. C's input i
   (which would be an integer)

and $H(C,i)$ outputs:

- "C halts", if C halts on i
- "C loops", if C loops on i.

Is there an *algorithm* A_H that computes the Halting Function H? Can we write such a program? Alternatively, is there a computer whose program is A_H? When you input the pair $\langle C,i\rangle$ to this computer, it should output "C halts" if another computer (namely, the computer for C) successfully halts and outputs a value when you give it input i. And it should output "C loops" if the computer for C goes into an infinite loop and never outputs any final answer. (C may, however, output some messages—e.g., "I'm still working; please be patient."—but it never halts with an answer to $C(i)$.)

The computer whose program is A_H is a kind of "super"-computer that takes as input, not only an integer i, but also another *computer* C. When you run A_H, it

[41] This sidebar is adapted from Rapaport 2023a, §7.7. For a simpler overview, see Brubaker 2023.

first feeds i to C, and then A_H runs C. If A_H detects that C has successfully output a value, then A_H outputs "C halts"; otherwise, A_H outputs "C loops".

It is important to be clear that it *can* be possible to write a program that will check if *another* program has an infinite loop. In other words, given a program C_1, there might be another program H_1 that will check whether C_1—*but not necessarily any other program*—has an infinite loop. What *cannot* be done is this: To write a single program A_H that will take *any program C whatsoever* and tell you whether C will halt or not.

Note, by the way, that we can't answer the question whether C halts on i by just running C on i: If it halts, we will know that it halts. But if it loops, how would we know that it loops? After all, it might just be taking a long time to halt.

There are two ways that we might try to write A_H.

1. You can imagine that $A_H(C,i)$ works as follows:

   - A_H gives C its input i, and then runs C on i.
   - If C halts on i, then A_H outputs "C halts";
     otherwise, A_H outputs "C loops".

   So, we might write A_H as follows:

   ```
   algorithm $A_H^1(C,i)$:
   begin
     if $C(i)$ halts
       then output '$C$ halts'
       else output '$C$ loops'
   end.
   ```

   This matches our definition of function H.

2. Here's another way to write A_H:

   ```
   algorithm $A_H^2(C,i)$:
   begin
     output '$C$ loops'; {i.e., make an initial guess that $C$ loops}
     if $C(i)$ halts
       then output '$C$ halts'; {i.e., revise your guess}
   end.
   ```

   But A_H^2 doesn't really do the required job: It doesn't give us a *definitive* answer to the question of whether C halts, because its initial answer is not really "C loops", but something like "C hasn't halted yet".[42] In any case, this won't work here, because we're going to need to convert our program for H to another program called A_H^*, and A_H^2 can't be converted that way, as we'll see.

The answer to our question about whether algorithm A_H exists or can be written is negative: There is no program for $H(C,i)$. In other words, $H(C,i)$ is a non-computable function. Note that it *is* a function: There exists a perfectly good set of input-output pairs that satisfies the extensional definition of 'function' and that looks like this:

$$\{\langle\langle C_1, i_1\rangle, \text{“}C_1 \text{ halts”}\rangle\ ,\ \ldots\ ,\langle\langle C_j, i_k\rangle, \text{“}C_j \text{ loops”}\rangle\ ,\ \ldots\}$$

The next section sketches a proof that H is not computable. The proof takes the form of a "*reductio ad absurdum*" argument. (See Sidebar E: *Reductio* Arguments.) So, for our proof, we will assume that H *is* computable, and derive a contradiction.

F.2 Proof Sketch that H Is Not Computable

F.2.1 Step 1

Assume that function H *is* computable.
So, there is an *algorithm* A_H that computes *function* H.
Now consider another algorithm, A_H^*, that works as follows:
A_H^* is just like algorithm A_H, except that:

- if C *halts* on i, then A_H^* *loops*

  (Remember: If C halts on i, then, by C's definition, A_H does *not* loop, because A_H outputs "C halts" and then halts.)

  and

- if C *loops* on i, then A_H^* outputs "C loops" and halts (just like A_H does).

Here is how we might write A_H^*, corresponding to the version of A_H that we called 'A_H^1' above:

```
algorithm $A_H^{1*}(C,i)$:
begin
  if $C(i)$ halts
```

[42]It is, in fact a trial-and-error program of the sort mentioned in §B.2.

```
        then while true do begin end
        else output ‘$C$ loops’
end.
```

Here, ‘true’ is a Boolean test that is always true, so the while-clause is an infinite loop.

Note that we cannot write a version of A^2_H that might look like this:

```
algorithm $A^{2*}_H(C,i)$:
begin
    output ‘$C$ loops’; {i.e., make an initial guess that $C$ loops}
    if $C(i)$ halts
        then while true do begin end {that is, if $C$ halts, then loop}
end.
```

Why not? Because if *C halts*, then the only output we will ever see is the message that says that *C loops*! That initial, incorrect guess is never revised. So, we'll stick with A_H (that is, with A^1_H) and with A^*_H (that is, with A^{1*}_H).

Note that if A_H exists, so does A^*_H. That is, we can turn A_H into A^*_H as follows: If A_H were to output "C halts", then let A^*_H go into an infinite loop. That is, replace A_H's "output ‘C halts’ " by A^*_H's infinite loop. This is important, because we are going to show that, in fact, A^*_H does *not* exist; hence, neither does A_H.

F.2.2 Step 2

Returning to our proof sketch, the next step is to code C as a number, so that it can be treated as input to itself. This is the way to simulate the idea of putting the C "machine" into the A_H machine and then having the A_H machine run the C machine.

So, how do you "code" a program as a number? This is an insight due to Gödel. To code *any* text (including a computer program) as a number in such a way that you could also *de*code it, begin by coding each symbol in the text as a unique number (for example, using the ASCII code). Suppose that these numbers, in order, are $L_1, L_2, L_3, \ldots, L_n$, where L_1 codes the first symbol in the text, L_2 codes the second, ..., and L_n codes the last symbol.

Then compute the following number:

$$2^{L_1} \times 3^{L_2} \times 5^{L_3} \times 7^{L_4} \times \ldots \times p_n^{L_n}$$

where p_n is the nth prime number, and where the ith factor in this product is the ith prime number raised to the L_ith power.

By the "Fundamental Theorem of Arithmetic",[43] the number that is the value of this product can be uniquely factored, so those exponents can be recovered, and then they can be decoded to yield the original text.[44]

F.2.3 Step 3

Now consider $A^*_H(C,C)$. This step is called "diagonalization". It looks like a form of self-reference, because it looks as if we are letting C take itself as input to itself—but actually C will take its own Gödel number as input. That is, suppose that you (1) code up program C as a Gödel number, (2) use it as input to the program C itself (after all, the Gödel number of C is an integer, and thus it is in the domain of the function that C computes, so it is a legal input for C), and (3) then let A^*_H do its job on *that* pair of inputs.

By the definition of A^*_H:

> if program C *halts* on input C, then $A^*_H(C,C)$ *loops*;
>
> and
>
> if program C *loops* on input C, then $A^*_H(C,C)$ *halts* and outputs "C loops".

F.2.4 Step 4

> The reason, essentially, is that the scope of the task covers *all* computer programs, including the program of the very computer attempting the task.
> —B. Jack Copeland (2017a, p. 61)

Now code A^*_H by a Gödel number! And consider $A^*_H(A^*_H,\ A^*_H)$. This is another instance of diagonalization. Again, it may look like some kind of self-reference, but it really isn't, because the first occurrence of 'A^*_H' names an algorithm, but the second and third occurrences are just numbers that happen to be the code for that algorithm.[45]

In other words, (1) code up A^*_H by a Gödel number, (2) use it as input to the program A^*_H itself, and then (3) let A^*_H do its job on that pair of inputs.

Again, by the definition of A^*_H:

[43] http://mathworld.wolfram.com/FundamentalTheoremofArithmetic.html

[44] Gödel numbering is actually a bit more complicated than this. For more information, see http://mathworld.wolfram.com/GoedelNumber.html or http://en.wikipedia.org/wiki/Godel_numbering.

[45] My notation here, cumbersome as it is(!), is nonetheless rather informal, but—I hope—clearer than it would be if I tried to be even more formally precise.

if program A_H^* *halts* on input A_H^*, then $A_H^*(A_H^*, A_H^*)$ *loops*;

and

if program A_H^* *loops* on input A_H^*, then $A_H^*(A_H^*, A_H^*)$ *halts* and outputs "A_H^* loops".

F.2.5 Final Result

But A_H^* outputting "A_H^* loops" means that A_H^* halts!

So, if A_H^* *halts* (outputting "A_H^* loops"), then it *loops*, and, if A_H^* *loops*, then it *halts*. In other words, *it loops if and only if it halts*; that is, it *does* loop if and only if it does *not* loop!

But that's a contradiction!

So, there is no such program as A_H^*. But that means that there is no such program as A_H. In other words, *the Halting Function H is not computable.*

G Glossary

Note to co-author and reviewers: Are there other terms that should be in this glossary?

Algorithm Roughly, a set of very explicit instructions (leaving no room for interpretation) for accomplishing a task. See §2.4.2.

Artificial Intelligence (AI) As used in this chapter, ‘AI’ refers to the computational study of cognition. See §2.

Chinese Room Argument John Searle’s thought experiment designed to show that a computer executing a Chinese natural-language-understanding program would not thereby understand Chinese, because a human who did not understand Chinese and who followed the program would still not understand Chinese. See Sidebar A.

Cognitive Science The interdisciplinary study of cognition. The central disciplines include (but are not limited to) computer science (especially AI), linguistics, philosophy, and psychology. Others include cognitive anthropology and cognitive neuroscience. See Rapaport 2000a.

Computable A mathematical function (or any task that can be expressed as such a function) is computable if there is an algorithm for calculating the outputs of the function (or for accomplishing the task). See §2.4.3 and Sidebar B.

Computer Science As used in this chapter, computer science is the scientific study of which problems can be solved, which tasks can be accomplished, and which features of the world can be understood computationally, and then to provide algorithms to show how this can be done efficiently, practically, physically, and ethically. See §2.4.1.

Hypercomputation A controversial and general form of computation that is said to be more powerful than Turing Machine computation. See §5.4.2.

First-Order Logic Sometimes called **predicate logic** or **predicate calculus**. The study of inferences that can be made on the basis of an analysis of atomic sentences (i.e., sentences without connectives such as ‘not’, ‘and’, ‘or’, etc.) into “terms” (noun phrases), “predicates” (verb phrases), and quantifiers (‘for all’, ‘there is’). What makes it “first-order”, roughly, is that only terms, and not predicates, can be quantified over. See, e.g., Rapaport 1992a,b.

Ineffable A phenomenon is ineffable if it cannot be expressed in language. Examples of possibly ineffable phenomena relevant to AI are implicit (or tacit) knowledge (“intuition”, “gut feelings”) (see §4.2.1) and qualia (see §5.3).

Halting Problem A version of a theorem proved in Turing 1936 to the effect that there is no algorithm (i.e., no computer program) H such that, for any algorithm (i.e., computer program) C (including H), H can determine whether or not C halts. See Sidebar F.

Intentionality As used in this chapter, a technical philosophical term for the property of being about, or being directed to, something. E.g., the word 'dog' is about dogs, the sentence 'Santa Claus is thin' is about Santa Claus, and I am currently thinking about intentionality, etc. See §2.3.1.

NP and NP-Complete Problems Computable problems that are among the most difficult to program efficiently. See Sidebar B.3.

Peano's Axioms Informally, the set $\mathbb{N}$ of natural numbers = $\{0, 1, 2, \ldots\}$. They are the numbers defined (recursively!) by *Peano's axioms*.

P1 Base case: $0 \in \mathbb{N}$. (That is, 0 is a natural number.)

P2 Recursive case: If $n \in \mathbb{N}$, then $S(n) \in \mathbb{N}$, where S is a one-to-one function from $\mathbb{N}$ to $\mathbb{N}$ such that $(\forall n \in \mathbb{N})[S(n) \neq 0]$.

$S(n)$ is called "the *successor* of n". So, the recursive case says that every natural number has a successor that is also a natural number. The fact that S is a *function* means that each $n \in \mathbb{N}$ has only *one* successor. The fact that S is *one-to-one* means that no two natural numbers have the *same* successor. And the fact that 0 is not the successor of any natural number means both that S is not an "onto" function and that 0 is the "first" natural number.

P3 Closure clause: Nothing else is a natural number.

We now have a set of natural numbers:

$$\mathbb{N} = \{0,\ S(0),\ S(S(0)),\ S(S(S(0))),\ \ldots\}$$

and, as is usually done, we define $1 =_{def} S(0)$, $2 =_{def} S(S(0))$, and so on. The closure clause guarantees that there are no other natural numbers besides 0 and its successors: Suppose that there *were* an $m \in \mathbb{N}$ that was neither 0 nor a successor of 0, nor a successor of any of 0's successors; without the closure clause, such an m could be used to start a "second" natural-number sequence: m, $S(m)$, $S(S(m))$, ... So, the closure clause ensures that no proper *superset* of $\mathbb{N}$ is also a set of natural numbers. Thus, in a sense, $\mathbb{N}$ is "bounded from above". But we also want to "bound" it from *below*; that is, we want to say that $\mathbb{N}$ is the *smallest* set satisfying (P1)–(P3). We do that with one more axiom:

P4 Consider an allegedly proper (hence, smaller) subset $\mathbf{M}$ of $\mathbb{N}$.
Suppose that:
(1) $0 \in \mathbf{M}$, and (2) for all $n \in \mathbb{N}$, if $n \in \mathbf{M}$, then $S(n) \in \mathbf{M}$.
Then $\mathbf{M} = \mathbb{N}$.

Stated thus, (P4) is the axiom that underlies the logical rule of inference known as "mathematical induction":

From the fact that 0 has a certain property **M**

and from the fact that any natural number that has the property **M** is such that its successor also has that property,

then it may be inferred that *all* natural numbers have that property.

See https://mathworld.wolfram.com/PeanosAxioms.html and Rapaport 2023a, §7.6.1.

Traveling Salesperson Problem The problem of finding the shortest possible route among a finite set of cities, visiting each one (except the first) only once and returning to the first one. It is *NP*-complete.

Turing Machine Not a real, physical machine, but an abstract, mathematical model of computation developed by Alan Turing in 1936, consisting of an arbitrarily large "tape" whose "cells" (like those of a spreadsheet) contain only '0' or '1', and a read-write head that can move to any cell and rewrite its contents. See §B.2.2.

Turing Test Alan Turing's 1950 thought experiment about the cognitive behavior of a computer. It was designed to replace the question "Can a machine think?". See §3.2.2.

Universal Turing Machine A Turing Machine that can have the program for any other Turing Machine encoded on its tape, which it can execute by explicitly following those instructions. It is "universal" in the sense that it can do what any other Turing Machine can do. Real physical computers that are programmable are implementations of Universal Turing Machines. See Sidebar B.2.3.

References

Aaronson, S. (2008, March). The limits of quantum computers. *Scientific American*, 62–69. http://www.cs.virginia.edu/~robins/The_Limits_of_Quantum_Computers.pdf.

Aaronson, S. (2011, 9 December). Quantum computing promises new insights, not just supermachines. *New York Times*, D5. 9 December; http://www.nytimes.com/2011/12/06/science/scott-aaronson-quantum-computing-promises-new-insights.html.

Aaronson, S. (2013). Why philosophers should care about computational complexity. In B. J. Copeland, C. J. Posy, and O. Shagrir (Eds.), *Computability: Turing, Gödel, Church, and Beyond*, pp. 261–327. Cambridge, MA: MIT Press. http://www.scottaaronson.com/papers/philos.pdf.

Aaronson, S. (2014, July-August). Quantum randomness. *American Scientist 102*(4), 266–271. http://www.americanscientist.org/issues/pub/quantum-randomness.

Aaronson, S. (2016). The ghost in the quantum Turing machine. In S. B. Cooper and A. Hodges (Eds.), *The Once and Future Turing*, pp. 193–296. Cambridge, UK: Cambridge University Press.

Aaronson, S. and B. Barak (2023, 27 April). Five worlds of AI. *Shtetel-Optimized*. https://scottaaronson.blog/?p=7266.

Adleman, L. M. (1998, August). Computing with DNA. *Scientific American*, 54–61. http://152.2.128.56/~montek/teaching/Comp790-Fall11/Home/Home_files/Adleman-ScAm94.pdf.

Aizawa, K. (2010, September). Computation in cognitive science: It is not all about Turing-equivalent computation. *Studies in History and Philosophy of Science 41*(3), 227–236.

Akman, V. and P. Blackburn (Eds.) (2000). *Alan Turing and Artificial Intelligence*. Special issue of *Journal of Logic, Language and Information* 9(4) (October). Table of contents at https://link.springer.com/journal/10849/9/4.

Alama, J. and J. Korbmacher (2021). The Lambda Calculus. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Summer 2021 ed.). Metaphysics Research Lab, Stanford University.

Anderson, A. R. and N. D. Belnap, Jr. (Eds.) (1975). *Entailment: The Logic of Relevance and Necessity*, Volume I. Princeton, NJ: Princeton University Press.

Anderson, A. R., N. D. Belnap, Jr., and J. M. Dunn (Eds.) (1992). *Entailment: The Logic of Relevance and Necessity*, Volume II. Princeton, NJ: Princeton University Press.

Arden, B. W. (Ed.) (1980). *What Can Be Automated? The Computer Science and Engineering Research Study (COSERS)*. Cambridge, MA: MIT Press.

Argamon, S., M. Koppel, J. Fine, and A. R. Shimoni (2003). Gender, genre, and writing style in formal written texts. *Text & Talk 23*(3), 321–346. http://writerunboxed.com/wp-content/uploads/2007/10/male-female-text-final.pdf.

Arvan, M. and C. J. Maley (2022). Panpsychism and AI consciousness. *Synthese 200*(244). https://doi.org/10.1007/s11229-022-03695-x.

Asimov, I. (1972). *The Gods Themselves*. Garden City, NY: Doubleday.

Asimov, I. (1987). *Fantastic Voyage II: Destination Brain*. New York: Doubleday.

Austin, A. K. (1983). An elementary approach to *NP*-completeness. *American Mathematical Monthly 90*, 398–399.

Awad, E., S. Levine, et al. (2022, May). Computational ethics. *Trends in Cognitive Sciences 26*(5), 388–404. https://doi.org/10.1016/j.tics.2022.02.009.

Bacon, D. (2010, December). What is computation? Computation and fundamental physics. *Ubiquity 2010*. Article 4, http://ubiquity.acm.org/article.cfm?id=1920826.

Bacon, D. and W. van Dam (2010, February). Recent progress in quantum algorithms. *Communications of the ACM 53*(2), 84–93.

Benacerraf, P. (1967, January). God, the Devil, and Gödel. *The Monist 51*(1), 9–32. Reprint at http://www2.units.it/etica/2003_1/3_monographica.htm; see also Lucas 1968.

Bernhardt, C. (2016). *Turing's Vision: The Birth of Computer Science*. Cambridge, MA: MIT Press.

Berreby, D. (2020, 22 November). Can we make our robots less biased than we are? *New York Times*. https://www.nytimes.com/2020/11/22/science/artificial-intelligence-robots-racism-police.html.

Black, M. (1952). *Critical Thinking: An Introduction to Logic and Scientific Method, second edition*. Englewood Cliffs, NJ: Prentice-Hall.

Block, N. (1995). On a confusion about a function of consciousness. *Behavioral and Brain Sciences 18*, 227–287. http://www.nyu.edu/gsas/dept/philo/faculty/block/papers/1995_Function.pdf.

Block, N. (1999, Spring/Fall). Sexism, racism, ageism, and the nature of consciousness. *Philosophical Topics 26*(1/2), 39–70. http://www.nyu.edu/gsas/dept/philo/faculty/block/papers/1999_Sexism.pdf.

Blum, M. and L. Blum (2021). A theoretical computer science perspective on consciousness. *Journal of Artificial Intelligence and Consciousness 8*(1), 1–42. https://www.worldscientific.com/doi/reader/10.1142/S2705078521500028.

Boden, M. A. (1977). *Artificial Intelligence and Natural Man*. New York: Basic Books.

Boden, M. A. (1990). *The Creative Mind: Myths & Mechanisms*. New York: Basic Books.

Boden, M. A. (2018). *AI: A Very Short Introduction*. Oxford: Oxford University Press.

Bolden, C. (2016, September). Katherine Johnson, the NASA mathematician who advanced human rights with a slide rule and pencil. *Vanity Fair*. http://www.vanityfair.com/culture/2016/08/katherine-johnson-the-nasa-mathematician-who-advanced-human-rights.

Boolos, G. S. (1968, December). Review of Lucas 1961; Benacerraf 1967. *Journal of Symbolic Logic 33*(4), 613–615.

Boolos, G. S. and R. C. Jeffrey (1974). *Computability and Logic*. Cambridge, UK: Cambridge University Press.

Bowles, N. (2019, 1 April). A journey—if you dare—into the minds of Silicon Valley programmers. *New York Times Book Review*. https://www.nytimes.com/2019/04/01/books/review/clive-thompson-coders.html.

Brachman, R. J. and H. J. Levesque (2022). *Machines Like Us: Toward AI with Common Sense*. Cambridge, MA: MIT Press.

Brentano, F. (1874). The distinction between mental and physical phenomena. In R. M. Chisholm (Ed.), *Realism and the Background of Phenomenology*, pp. 39–61. New York: Free Press, 1960. Trans. by D.B. Terrell.

Bringsjord, S. (2021, 22 March 2021). Ethical A.I. *The Academic Minute*. https://academicminute.org/2021/03/selmer-bringsjord-rensselaer-polytechnic-institute-ethical-a-i/.

Bringsjord, S. and N. S. Govindarajulu (2020). Artificial Intelligence. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Summer 2020 ed.). Metaphysics Research Lab, Stanford University.

Brooks, D. (2023, 2 February). In the age of A.I., major in being human. *New York Times*. https://www.nytimes.com/2023/02/02/opinion/ai-human-education.html.

Brown, T. B., B. Mann, et al. (2020, 22 July). Language models are few-shot learners. https://arxiv.org/abs/2005.14165.

Brubaker, B. (2023, 5 September). Alan Turing and the power of negative thinking. *Quanta Magazine*. https://www.quantamagazine.org/alan-turing-and-the-power-of-negative-thinking-20230905/.

Buckner, C. (2013). Morgan's Canon, meet Hume's Dictum: Avoiding anthropofabulation in cross-species comparisons. *Biology & Philosophy 28*, 853–871. Preprint at http://philsci-archive.pitt.edu/16327/.

Buckner, C. (2020). Black boxes or unflattering mirrors? Comparative bias in the science of machine behavior. *British Journal for the Philosophy of Science*. Preprint at http://cameronbuckner.net/professional/blackboxesormirrors.pdf.

Buckner, C. J. (2024). *From Deep Learning to Rational Machines: What the History of Philosophy Can Teach Us about the Future of Artificial Intelligence*. New York: Oxford University Press.

Buechner, J. (2011). Not even computing machines can follow rules: Kripke's critique of functionalism. In A. Berger (Ed.), *Saul Kripke*, pp. 343–367. New York: Cambridge University Press.

Buechner, J. (2018, Spring). Does Kripke's argument against functionalism undermine the standard view of what computers are? *Minds and Machines 28*(3), 491–513.

Bundy, A. (2017, February). Smart machines are not a threat to humanity. *Communications of the ACM 60*(2), 40–42.

Burdick, A. (2020, 24 November). Artificial intelligence is here. *New York Times Science Times*. https://nims360.blogspot.com/2020/11/science-times-artificial-intelligence.html.

Byrne, A. (2020). Inverted qualia. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Spring 2020 ed.). Metaphysics Research Lab, Stanford University.

Čapek, K. (1920). R.U.R.: Rossom's Universal Robots. Trans. by Paul Selver and Nigel Playfair at http://preprints.readingroo.ms/RUR/rur.pdf; trans. by David Wyllie at http://ebooks.adelaide.edu.au/c/capek/karel/rur/.

Care, C. (2007, May). Not only digital: A review of ACM's early involvement with analog computing technology. *Communications of the ACM 50*(5), 42–45.

Carey, B. (2019, 26 June). 'It's gigantic': A new way to gauge the chances for unresponsive patients. *New York Times*. https://www.nytimes.com/2019/06/26/health/brain-injury-eeg-consciousness.html.

Casner, S. M., E. L. Hutchins, and D. Norman (2016, May). The challenges of partially automated driving. *Communications of the ACM 59*(5), 70–77. https://cacm.acm.org/magazines/2016/5/201592-the-challenges-of-partially-automated-driving/fulltext.

Cassenti, D. N., V. D. Veksler, and F. E. Ritter (2022). Editor's review and introduction: Cognition-inspired artificial intelligence. *Topics in Cognitive Science*. https://onlinelibrary.wiley.com/doi/abs/10.1111/tops.12622.

Chalmers, D. J. (1995). Facing up to the problem of consciousness. *Journal of Consciousness Studies 2*(3), 200–219. http://consc.net/papers/facing.html.

Chalmers, D. J. (1996). *The Conscious Mind: In Search of a Fundamental Theory*. New York: Oxford University Press. http://tinyurl.com/plv877.

Chalmers, D. J. (2010). The singularity: A philosophical analysis. *Journal of Consciousness Studies 17*, 7–65. http://consc.net/papers/singularity.pdf.

Chalmers, D. J. (2011, October-December). A computational foundation for the study of cognition. *Journal of Cognitive Science (South Korea) 12*(4), 323–357. http://cogsci.snu.ac.kr/jcs/issue/vol12/no4/01Chalmers.pdf.

Chalmers, D. J. (2012). The singularity: A reply. *Journal of Consciousness Studies 19*(7–8), 141–167. http://consc.net/papers/singreply.pdf.

Charlesworth, A. (2014). The Comprehensibility Theorem and the foundations of artificial intelligence. *Minds and Machines 24*, 439–476. https://link.springer.com/content/pdf/10.1007/s11023-014-9349-3.pdf.

Chiang, T. (2002). Seventy-two letters. In *Stories of Your Life and Others*, pp. 147–200. New York: Vintage. https://tinyurl.com/chiang2002.

Chirimuuta, M. (2021). Your brain is like a computer: Function, analogy, simplification. In F. Calzavarini and M. Viola (Eds.), *Neural Mechanisms: New Challenges in the Philosophy of Neuroscience*, pp. 235–261. Cham, Switzerland: Springer. https://ruccs.rutgers.edu/images/talks-materials/Chirimuuta_Brain-Computer-Analogy_Rutgers.pdf.

Chomsky, N. (2017). The Galilean challenge. *Inference: International Review of Science 3*(1). http://inference-review.com/article/the-galilean-challenge.

Chow, S. J. (2015). Many meanings of 'heuristic'. *British Journal for the Philosophy of Science 66*, 977–1016.

Church, A. (1936, March). A note on the Entscheidungsproblem. *Journal of Symbolic Logic 1*(1), 40–41. See also "Correction to *A Note on the Entscheidungsproblem*", in *Journal of Symbolic Logic* 1(3) (September): 101–102.

Church, A. (1937, March). Review of Turing 1936. *Journal of Symbolic Logic 2*(1), 42–43. For commentary on this review, see Hodges 2013.

Clark, K. L. and D. F. Cowell (1976). *Programs, Machines, and Computation: An Introduction to the Theory of Computing*. London: McGraw-Hill.

Cole, D. (2014, 31 December). Alan Turing & the Chinese Room Argument. http://www.thecritique.com/articles/alan-turing-the-chinese-room-argument/.

Cole, D. (2020). The Chinese Room Argument. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Spring 2020 ed.). Metaphysics Research Lab, Stanford University.

Cook, S. A. (1983, June). An overview of computational complexity. *Communications of the ACM 26*(6), 400–408. http://www.jdl.ac.cn/turing/pdf/p400-cook.pdf.

Copeland, B. J. (1997, May). The broad conception of computation. *American Behavioral Scientist 40*(6), 690–716.

Copeland, B. J. (2002a, November). Hypercomputation. *Minds and Machines 12*(4), 461–502.

Copeland, B. J. (Ed.) (2002b). *Hypercomputation*. Special issue of *Minds and Machines* 12(4) (November). http://link.springer.com/journal/11023/12/4/.

Copeland, B. J. (Ed.) (2003). *Hypercomputation (continued)*. Special issue of *Minds and Machines* 13(1) (February). http://link.springer.com/journal/11023/13/1/.

Copeland, B. J. (Ed.) (2004). *The Essential Turing*. Oxford: Oxford University Press.

Copeland, B. J. (2017a). Hilbert and his famous problem. In B. J. Copeland, J. P. Bowen, M. Sprevak, and R. Wilson (Eds.), *The Turing Guide*, pp. 57–65. Oxford: Oxford University Press.

Copeland, B. J. (2017b). Intelligent machinery. In B. J. Copeland, J. P. Bowen, M. Sprevak, and R. Wilson (Eds.), *The Turing Guide*, pp. 265–275. Oxford: Oxford University Press.

Copeland, B. J. (2023, July/September). Early AI in Britain: Turing et al. *IEEE Annals of the History of Computing 45*, 19–31. https://doi.ieeecomputersociety.org/10.1109/MAHC.2023.3300660.

Copeland, B. J. and J. Bowen (2017). Life and work. In B. J. Copeland, J. P. Bowen, M. Sprevak, and R. Wilson (Eds.), *The Turing Guide*, pp. 3–17. Oxford: Oxford University Press.

Copeland, B. J. and D. Prinz (2017). Computer chess—the first moments. In B. J. Copeland, J. Bowen, M. Sprevak, and R. Wilson (Eds.), *The Turing Guide*, pp. 327–346. Oxford: Oxford University Press.

Copeland, B. J. and D. Proudfoot (2017). Connectionism: Computing with neurons. In B. J. Copeland, J. Bowen, M. Sprevak, and R. Wilson (Eds.), *The Turing Guide*, pp. 309–314. Oxford: Oxford University Press.

Copeland, B. J. and O. Shagrir (2013). Turing versus Gödel on computability and the mind. In B. J. Copeland, C. J. Posy, and O. Shagrir (Eds.), *Computability: Turing, Gödel, Church, and Beyond*, pp. 1–33. Cambridge, MA: MIT Press.

Copeland, B. J. and O. Shagrir (2019, January). The Church-Turing thesis: Logical limit or breachable barrier? *Communications of the ACM 62*(1), 66–74. https://cacm.acm.org/magazines/2019/1/233526-the-church-turing-thesis/fulltext.

Corballis, M. C. (2007, May-June). The uniqueness of human recursive thinking. *American Scientist 95*, 240–248.

Corry, L. (2017, August). Turing's pre-war analog computers: The fatherhood of the modern computer revisited. *Communications of the ACM 60*(8), 50–58. https://cacm.acm.org/magazines/2017/8/219602-turings-pre-war-analog-computers/fulltext.

Crawford, K. (2016, 25 June). Artificial intelligence's white guy problem. *New York Times*. https://www.nytimes.com/2016/06/26/opinion/sunday/artificial-intelligences-white-guy-problem.html.

Crosby, M. (2020, December). Building thinking machines by solving animal cognition tasks. *Minds and Machines 30*(4), 589–615. https://link.springer.com/article/10.1007/s11023-020-09535-6.

Cummins, R. and G. Schwarz (1991). Connectionism, computation, and cognition. In T. E. Horgan and J. L. Tienson (Eds.), *Connectionism and the Philosophy of Mind*, pp. 60–73. Dordrecht, The Netherlands: Kluwer.

Damassino, N. and N. Novelli (Eds.) (2020). *Rethinking, Reworking and Revolutionising the Turing Test*. Special Issue of *Minds and Machines*, Vol. 30, No. 4 (December). https://link.springer.com/journal/11023/volumes-and-issues/30-4.

Darwiche, A. (2018, October). Human-level intelligence or animal-like abilities? *Communications of the ACM 61*(10), 56–67. https://arxiv.org/abs/1707.04327.

Davidson, D. (1990). Turing's Test. In K. Said, M. Said, W. Newton-Smith, R. Viale, and K. Wilkes (Eds.), *Modelling the Mind*, pp. 1–11. Oxford: Clarendon Press.

Davidson, J. (2006, 21 August). Measure for measure: Exploring the mysteries of conducting. *The New Yorker*, 60–69.

Davis, E. (2015). Ethical guidelines for a superintelligence. *Artificial Intelligence 220*, 121–124. https://www.cs.nyu.edu/davise/papers/Bostrom.pdf.

Davis, M. D. (Ed.) (1965). *The Undecidable: Basic Papers on Undecidable Propositions, Unsolvable Problems and Computable Functions*. New York: Raven Press.

Davis, M. D. (1990). Is mathematical insight algorithmic? *Behavioral and Brain Sciences 13*(4), 659–660. http://www.cs.nyu.edu/faculty/davism/penrose.ps.

Davis, M. D. (1993). How subtle is Gödel's theorem? More on Roger Penrose. *Behavioral and Brain Sciences 16*(3), 611. http://www.cs.nyu.edu/faculty/davism/penrose2.ps.

Davis, M. D. (2004). The myth of hypercomputation. In C. Teuscher (Ed.), *Alan Turing: The Life and Legacy of a Great Thinker*, pp. 195–212. Berlin: Springer. http://www1.maths.leeds.ac.uk/~pmt6sbc/docs/davis.myth.pdf.

Davis, M. D. (2006). Why there is no such discipline as hypercomputation. *Applied Mathematics and Computation 178*, 4–7.

de Souza, N. (2021, 20 April). Editing humanity's future. *New York Review of Books 68*(7), 21–23. https://www.nybooks.com/articles/2021/04/29/crispr-editing-humanity-future/.

Dedekind, R. (1890). Letter to Keferstein. In J. van Heijenoort (Ed.), *From Frege to Gödel: A Source Book in Mathematical Logic, 1879–1931*, pp. 98–103. Cambridge, MA: Harvard University Press. Trans. by Hao Wang and Stefan Bauer-Mengelberg.

Dehaene, S., H. Lau, and S. Kouider (2017, 27 October). What is consciousness, and could machines have it? *Science 358*(6362), 486–492. https://science.sciencemag.org/content/358/6362/486. See correspondence, *Science*, Vol. 359, No. 6374 (26 January 2018): 400–402.

Dehaene, S., H. Lau, and S. Kouider (2018, 26 January). Response. *Science 359*(6374), 400–402.

Dennett, D. C. (1971). Intentional systems. *Journal of Philosophy 68*, 87–106. Reprinted in Daniel C. Dennett, *Brainstorms* (Montgomery, VT: Bradford Books, 1978): 3–22.

Dennett, D. C. (1975). Why the law of effect will not go away. *Journal for the Theory of Social Behaviour 5*(2), 169–188. Reprinted in Daniel C. Dennett, *Brainstorms* (Montgomery, VT: Bradford Books): 71–89, https://dl.tufts.edu/downloads/j9602b715?filename=2r36v862p.pdf.

Dennett, D. C. (1978, July). Why you can't make a computer feel pain. *Synthese 38*(3), 415–456. https://dl.tufts.edu/downloads/9w032f88p?filename=m039kh27m.pdf. Reprinted in Daniel C. Dennett, *Brainstorms* (Montgomery, VT: Bradford Books, 1978): 190–229.

Dennett, D. C. (1979). Artificial Intelligence as philosophy and as psychology. In M. D. Ringle (Ed.), *Philosophical Perspectives in Artificial Intelligence*, pp. 57–78. Brighton, Sussex, UK: Harvester Press.

Dennett, D. C. (1985). Can machines think? In M. Shafto (Ed.), *How We Know*, pp. 121–145. San Francisco: Harper & Row. https://tinyurl.com/dennett1985.

Dennett, D. C. (1987). *The Intentional Stance*. Cambridge, MA: MIT Press.

Dennett, D. C. (1992). *Consciousness Explained*. Boston: Little, Brown and Co.

Dennett, D. C. (1995). *Darwin's Dangerous Idea*. New York: Simon & Schuster.

Dennett, D. C. (2009a, 16 June). Darwin's 'strange inversion of reasoning'. *Proceedings of the National Academy of Science 106, suppl. 1*, 10061–10065. http://www.pnas.org/cgi/doi/10.1073/pnas.0904433106. See also Dennett 2013b.

Dennett, D. C. (2009b). Intentional systems theory. In B. McLaughlin, A. Beckermann, and S. Walter (Eds.), *The Oxford Handbook of Philosophy of Mind*, pp. 339–350. Oxford: Oxford University Press. https://ase.tufts.edu/cogstud/dennett/papers/intentionalsystems.pdf.

Dennett, D. C. (2013a). *Intuition Pumps and Other Tools for Thinking*. New York: W.W. Norton.

Dennett, D. C. (2013b). Turing's 'strange inversion of reasoning'. In S. B. Cooper and J. van Leeuwen (Eds.), *Alan Turing: His Work and Impact*, pp. 569–573. Amsterdam: Elsevier. See also Dennett 2009a.

Dennett, D. C. (2017). *From Bacteria to Bach and Back: The Evolution of Mind*. New York: W.W. Norton.

Dershowitz, N. and Y. Gurevich (2008, September). A natural axiomatization of computability and proof of Church's thesis. *Bulletin of Symbolic Logic 14*(3), 299–350. http://research.microsoft.com/pubs/70459/tr-2007-85.pdf.

Descartes, R. (1637). Discourse on method. In E. S. Haldane and G. Ross (Eds.), *The Philosophical Works of Descartes*, pp. 79–130. Cambridge, UK: Cambridge University Press, 1970. Other online versions at https://www.gutenberg.org/files/59/59-h/59-h.htm and https://www.earlymoderntexts.com/assets/pdfs/descartes1637.pdf.

Descartes, R. (1641). Meditations on first philosophy. In E. S. Haldane and G. Ross (Eds.), *The Philosophical Works of Descartes*, pp. 131–199. Cambridge, UK: Cambridge University Press, 1970. Online versions at http://selfpace.uconn.edu/class/percep/DescartesMeditations.pdf and https://www.earlymoderntexts.com/assets/pdfs/descartes1641_3.pdf.

Dietrich, E. (1993). The ubiquity of computation. *Think 2*, 12–78. Reprinted in *Psycoloquy* 12(040) (2001), Art. 7, http://www.cogsci.ecs.soton.ac.uk/cgi/psyc/newpsy?12.040.

Dietrich, E., C. Fields, J. P. Sullins, B. van Heuveln, and R. Zebrowski (2021). *Great Philosophical Objections to Artificial Intelligence: The History and Legacy of the AI Wars*. London: Bloomsbury Academic.

Dipert, R. R. (1993). *Artifacts, Art Works, and Agency*. Philadelphia: Temple University Press.

Dreyfus, H. L. (1965). *Alchemy and Artificial Intelligence*. Santa Monica, CA: Rand Corporation. https://www.rand.org/pubs/papers/P3244.html.

Dreyfus, H. L. (1992). *What Computers Still Can't Do: A Critique of Artificial Reason*. Cambridge, MA: MIT Press.

Dreyfus, H. L. and S. E. Dreyfus (1986). *Mind over Machine: The Power of Human Intuition and Expertise in the Era of the Computer*. New York: Free Press.

Dummett, M. A. (1976). What is a theory of meaning? (II). In G. Evans and J. McDowell (Eds.), *Truth and Meaning: Essays in Semantics*, pp. 67–137. Oxford: Clarendon Press.

Durán, J. M. (2020). What is a simulation model? *Minds and Machines 30*, 301–323. https://link.springer.com/content/pdf/10.1007/s11023-020-09520-z.pdf.

Duwell, A. (2021). *Physics and Computation*. Cambridge, UK: Cambridge University Press. https://doi.org/10.1017/9781009104975.

Dyson, G. (2012, 23 February). Turing centenary: The dawn of computing. *Nature 482*(7386), 459–460. doi:10.1038/482459a.

Ebcioğlu, K. (1990). An expert system for harmonizing chorales in the style of J.S. Bach. *Journal of Logic Programming 8*, 145–185. https://tinyurl.com/yy5wz8jt.

Edelman, S. (2008). *Computing the Mind*. New York: Oxford University Press.

Edelman, S. (2011, 17 March). Regarding reality: Some consequences of two incapacities. *Frontiers in Psychology 2*(44), 1–8. https://www.frontiersin.org/articles/10.3389/fpsyg.2011.00044/full.

Eden, A. H., J. H. Moor, J. H. Søraker, and E. Steinhart (Eds.) (2012). *Singularity Hypotheses: A Scientific and Philosophical Assessment*. Berlin: Springer. http://singularityhypothesis.blogspot.co.uk/.

Feferman, S. (2009, April). Gödel, nagel, minds, and machines. *Journal of Philosophy 106*(4), 201–219.

Feferman, S. (2011). Gödel's incompleteness theorems, free will, and mathematical thought. In R. Swinburne (Ed.), *Free Will and Modern Science*. Oxford University Press/British Academy. https://philarchive.org/archive/FEFGIT.

Fetzer, J. H. (2011, January-June). Minds and machines: Limits to simulations of thought and action. *International Journal of Signs and Semiotic Systems 1*(1), 39–48.

Findler, N. V. (1993). Heuristic. In A. Ralston and E. D. Reilly (Eds.), *Encyclopedia of Computer Science, 3rd Edition*, pp. 611–612. New York: Van Nostrand Reinhold.

Fitch, W. T., M. D. Hauser, and N. Chomsky (2005). The evolution of the language faculty: Clarifications and implications. *Cognition 97*, 179–210. http://dash.harvard.edu/bitstream/handle/1/3117935/Hauser_EvolutionLanguageFaculty.pdf.

Floridi, L. and M. Chiriatti (2020, December). GPT-3: Its nature, scope, limits, and consequences. *Minds and Machines 30*(4), 681–694. https://link.springer.com/article/10.1007/s11023-020-09548-1.

Fodor, J. A. (1974, October). Special sciences (or: The disunity of science as a working hypothesis). *Synthese 28*(2), 97–115.

Folina, J. (1998). Church's thesis: Prelude to a proof. *Philosophia Mathematica 6*, 302–323.

Fortnow, L. (2009, September). The status of the P versus NP problem. *Communications of the ACM 52*(9), 78–86. http://cacm.acm.org/magazines/2009/9/38904-the-status-of-the-p-versus-np-problem/fulltext.

Fortnow, L. (2010, December). What is computation? *Ubiquity 2010*. Article 5, http://ubiquity.acm.org/article.cfm?id=1921573.

Fortnow, L. (2013). *The Golden Ticket: P, NP, and the Search for the Impossible*. Princeton, NJ: Princeton University Press.

Fortnow, L. (2022, January). Fifty years of P vs. NP and the possibility of the impossible. *Communications of the ACM 65*(1), 76–85. https://cacm.acm.org/magazines/2022/1/257448-fifty-years-of-p-vs-np-and-the-possibility-of-the-impossible/fulltext.

Franzén, T. (2005). *Gödel's Theorem: An Incomplete Guide to Its Use and Abuse*. Wellesley, MA: A K Peters.

French, R. M. (2000). The Turing test: The first fifty years. *Trends in Cognitive Sciences 4*(3), 115–121. http://leadserv.u-bourgogne.fr/rfrench/french/TICS_turing.pdf.

Gandy, R. (1988). The confluence of ideas in 1936. In R. Herken (Ed.), *The Universal Turing Machine: A Half-Century Survey, Second Edition*, pp. 51–102. Vienna: Springer-Verlag. https://fi.ort.edu.uy/innovaportal/file/20124/1/41-herken_ed._95_-_the_universal_turing_machine.pdf.

Gardner, H. (1983). *Frames of Mind: The Theory of Multiple Intelligences*. New York: Basic Books, 2011.

Garnelo, M. and M. Shanahan (2019). Reconciling deep learning with symbolic artificial intelligence: Representing objects and relations. *Current Opinion in Behavioral Sciences 29*, 17–23. https://www.sciencedirect.com/science/article/pii/S2352154618301943.

Gleick, J. (2021, 29 April). Eclipsed by fame. *New York Review of Books 68*(7), 34–36. https://www.nybooks.com/articles/2021/04/29/stephen-hawking-eclipsed-by-fame/.

Goertzel, B. and C. Pennachin (Eds.) (2007). *Artificial General Intelligence*. Berlin: Springer Verlag.

Gold, E. M. (1967). Language identification in the limit. *Information and Control 10*, 447–474. https://www.sciencedirect.com/science/article/pii/S0019995867911655.

Goodman, N. D. (1984, July). The knowing mathematician. *Synthese 60*(1), 21–38.

Gopnik, A. (2009). *The Philosophical Baby: What Children's Minds Tell Us about Truth, Love, and the Meaning of Life*. New York: Farrar, Straus and Giroux.

Gorman, J. (2021, 7 September). How the cat gets its stripes: It's genetics, not a folk tale. *New York Times*. https://www.nytimes.com/2021/09/07/science/cat-stripes-genetics.html.

Griffiths, T. L. (2020, November). Understanding human intelligence through human limitations. *Trends in Cognitive Sciences 24*(11), 873–883. https://arxiv.org/pdf/2009.14050.pdf.

Grover, L. K. (1999, July/August). Quantum computing. *The Sciences*, 24–30. http://cryptome.org/qc-grover.htm.

Gunderson, K. (1964, July). Descartes, La Mettrie, language, and machines. *Philosophy 39*(149), 193–222.

Gurevich, Y. (1999, February). The sequential ASM thesis. *Bulletin of the European Association for Theoretical Computer Science 67*, 93–124. http://research.microsoft.com/en-us/um/people/gurevich/Opera/136.pdf.

Hadjeres, G., F. Pachet, and F. Nielsen (2017). DeepBach: A steerable model for Bach chorales generation. *Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, PMLR 70*. https://arxiv.org/pdf/1612.01010.pdf.

Halina, M. (2021). Insightful artificial intelligence. *Mind & Language 36*(2), 315–329. https://onlinelibrary.wiley.com/doi/10.1111/mila.12321.

Halverson, T., C. W. Myers, J. M. Gearhart, M. W. Linakis, and G. Gunzelmann (2022). Physiocognitive modeling: Explaining the effects of caffeine on fatigue. *Topics in Cognitive Science*. https://doi.org/10.1111/tops.12615.

Harnad, S. (1991). Other bodies, other minds: A machine incarnation of an old philosophical problem. *Minds and Machines 1*(1), 43–54. http://www.southampton.ac.uk/~harnad/Papers/Harnad/harnad91.otherminds.html.

Harris, P. (2000). *The Work of the Imagination*. Malden, MA: Blackwell.

Haugeland, J. (1981, Spring). Analog and analog. *Philosophical Topics 12*(1), 213–225.

Haugeland, J. (1985). *Artificial Intelligence: The Very Idea*. Cambridge, MA: MIT Press.

Hauser, L. (2001). Chinese room argument. *Internet Encyclopedia of Philosophy*. https://www.iep.utm.edu/chineser/.

Hauser, M. D., N. Chomsky, and W. T. Fitch (2002, 22 November). The faculty of language: What is it, who has it, and how did it evolve. *Science 298*, 1569–1579. http://www.chomsky.info/articles/20021122.pdf.

Hayes, B. (1995, July-August). The square root of NOT. *American Scientist 83*, 304–308. http://bit-player.org/wp-content/extras/bph-publications/AmSci-1995-07-Hayes-quantum.pdf.

Hayes, B. (2014, January-February). Programming your quantum computer. *American Scientist 102*(1), 22–25. https://www.americanscientist.org/article/programming-your-quantum-computer.

Henry, O. (1906a). The romance of a busy broker. In *The Complete Works of O. Henry*, pp. 67–69. Garden City, NY: Doubleday, Doran & Co. https://americanliterature.com/author/o-henry/short-story/the-romance-of-a-busy-broker.

Henry, O. (1906b). A service of love. In *The Complete Works of O. Henry*, pp. 20–24. Garden City, NY: Doubleday, Doran & Co. https://americanliterature.com/author/o-henry/short-story/a-service-of-love.

Hernández-Orallo, J. (2020, December). Twenty years beyond the Turing test: Moving beyond the human judges too. *Minds and Machines 30*(4), 533–562. https://link.springer.com/article/10.1007/s11023-020-09549-0.

Hilbert, D. (1900, July). Mathematical problems: Lecture delivered before the International Congress of Mathematicians at Paris in 1900. *Bulletin of the American Mathematical Society 8 (1902)*(10), 437–479. Trans. by Mary Winston Newson; first published in *Göttinger Nachrichten* (1900): 253–297.

Hilpinen, R. (2011). Artifact. In E. N. Zalta (Ed.), *Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab. http://plato.stanford.edu/entries/artifact/.

Hintikka, J. and A. Mutanen (1997). An alternative concept of computability. In J. Hintikka (Ed.), *Language, Truth, and Logic in Mathematics*, pp. 174–188. Dordrecht, The Netherlands: Springer.

Hodges, A. (2013). Church's review of computable numbers. In S. B. Cooper and J. van Leeuwen (Eds.), *Alan Turing: His Work and Impact*, pp. 117–118. Amsterdam: Elsevier.

Hofstadter, D. R. (1979). *Gödel, Escher, Bach: An Eternal Golden Braid*. New York: Basic Books.

Hofstadter, D. R. (1981, May). Metamagical themas: A coffeehouse conversation on the Turing test to determine if a machine can think. *Scientific American*, 15–36. https://cs.brynmawr.edu/~dblank/csem/coffeehouse.html. Reprinted as "The Turing Test: A Coffeehouse Conversation", in Douglas R. Hofstadter & Daniel C. Dennett (eds.), *The Mind's I: Fantasies and Reflections on Self and Soul* (New York: Basic Books, 1981): 69–95.

Hollan, J., E. Hutchins, and D. Kirsh (2000, June). Distributed cognition: Toward a new foundation for human-computer interaction research. *ACM Transactions on Computer-Human Interaction 7*(2), 174–196. https://www.lri.fr/~mbl/Stanford/CS477/papers/DistributedCognition-TOCHI.pdf.

Holst, P. A. (2000). Analog computer. In A. Ralston, E. D. Reilly, and D. Hemmendinger (Eds.), *Encyclopedia of Computer Science, 4th Edition*, pp. 53–59. New York: Grove's Dictionaries.

Homer, S. and A. L. Selman (2011). *Computability and Complexity Theory, 2nd Edition*. New York: Springer.

Horst, S. (2014, 14 December). The computational theory of mind: Alan Turing & the Cartesian challenge. http://www.thecritique.com/articles/the-computational-theory-of-mind-alan-turing-the-cartesian-challenge/.

Hutchins, E. (1995a). *Cognition in the Wild*. Cambridge, MA: MIT Press.

Hutchins, E. (1995b). How a cockpit remembers its speeds. *Cognitive Science 19*, 265–288.

Hutchins, E. (2010). Cognitive ecology. *Topics in Cognitive Science 2*, 705–715. https://onlinelibrary.wiley.com/doi/epdf/10.1111/j.1756-8765.2010.01089.x.

Irmak, N. (2012). Software is an abstract artifact. *Grazer Philosophische Studien 86*, 55–72. http://philpapers.org/archive/IRMSIA.pdf.

Ishiguro, K. (2021). *Klara and the Sun*. London: Faber.

Jackendoff, R. and S. Pinker (2005). The nature of the language faculty and its implications for evolution of language. *Cognition 97*, 211–225. http://pinker.wjh.harvard.edu/articles/papers/2005_09_Jackendoff_Pinker.pdf.

Jackson, A. S. (1960). *Analog Computation*. New York: McGraw-Hill.

Jackson, F. and P. Pettit (2002, December). Response-dependence without tears. *Philosophical Issues 12*(*Noûs* 36(s1)), 97–117.

Jacob, P. (2019). Intentionality. In E. N. Zalta (Ed.), *Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University. https://plato.stanford.edu/archives/win2019/entries/intentionality/.

Jarvis, B. (2021, 28 January). What can covid-19 teach us about the mysteries of smell? *New York Times Magazine*. https://www.nytimes.com/2021/01/28/magazine/covid-smell-science.html.

Jefferson, G. (1949, 25 June). The mind of mechanical man. *British Medical Journal 1*(4616), 1105–1110. The Lister Oration.

Johnson, D. G. and M. Verdicchio (2023, February). Ethical AI is not about AI. *Communications of the ACM 66*(2), 32–34.

Joyce, D. (2005). The Dedekind/Peano axioms. http://aleph0.clarku.edu/~djoyce/numbers/peano.pdf.

Kennedy, H. C. (1968, November). Giuseppe Peano at the University of Turin. *Mathematics Teacher*, 703–706. Reprinted in Kennedy, Hubert C. (2002), *Twelve Articles on Giuseppe Peano* (San Francisco: Peremptory Publications): 14–19, http://hubertkennedy.angelfire.com/TwelveArticles.pdf.

Kim, Q. S. (2015, 27 October). Stuart Russell on why moral philosophy will be big business in tech. *KQED: The California Report*. https://www.kqed.org/news/10734380/stuart-russell-on-a-i-and-how-moral-philosophy-will-be-big-business.

Kirk, R. (1974, January). Sentience and behaviour. *Mind 83*(329), 43–60.

Kirkpatrick, K. (2023, February). Can AI demonstrate creativity? *Communications of the ACM 66*(2), 21–23. https://cacm.acm.org/magazines/2023/2/268946-can-ai-demonstrate-creativity/fulltext.

Koellner, P. (2018, July). On the question of whether the mind can be mechanized, I: From Gödel to Penrose. *Journal of Philosophy 115*(7), 337–360.

Kolbert, E. (2022, 13 June). Contact. *The New Yorker*, 22–26. https://tinyurl.com/3u2mxjve.

Korf, R. E. (1992). Heuristics. In S. C. Shapiro (Ed.), *Encyclopedia of Artificial Intelligence, 2nd edition*, pp. 611–615. New York: John Wiley & Sons.

Kriegel, U. (2006). Consciousness: Phenomenal consciousness, access consciousness, and scientific practice. In P. Thagard (Ed.), *Handbook of Philosophy of Psychology and Cognitive Science*, pp. 195–217. Amsterdam: North-Holland. http://www.uriahkriegel.com/userfiles/downloads/concept.pdf.

Kriegel, U. (2016). Brentano's mature theory of intentionality. *Journal for the History of Analytical Philosophy 4*(2), 1–15. https://jhaponline.org/jhap/article/view/2428.

Kripke, S. A. (2013). The Church-Turing "thesis" as a special corollary of Gödel's completeness theorem. In B. J. Copeland, C. J. Posy, and O. Shagrir (Eds.), *Computability: Turing, Gödel, Church, and Beyond*, pp. 77–104. Cambridge, MA: MIT Press.

Kryven, M., T. D. Ullman, W. Cowan, and J. B. Tenenbaum (2021, September). Plans or outcomes: How do we attribute intelligence to others? *Cognitive Science 45*(9). https://doi.org/10.1111/cogs.13041.

Kugel, P. (2002, November). Computing machines can't be intelligent (. . . and Turing said so). *Minds and Machines 12*(4), 563–579. http://cs.bc.edu/~kugel/Publications/Hyper.pdf.

LaForte, G., P. J. Hayes, and K. M. Ford (1998). Why Gödel's theorem cannot refute computationalism. *Artificial Intelligence 104*, 265–286. https://www.researchgate.net/publication/222498386_Why_Godel's_theorem_cannot_refute_computationalism.

Lakoff, G. and M. Johnson (1980a). Conceptual metaphor in everyday language. *Journal of Philosophy 77*(8), 453–486.

Lakoff, G. and M. Johnson (1980b). *Metaphors We Live By*. Chicago: University of Chicago Press.

Landgrebe, J. and B. Smith (2019, 2 December). There is no artificial general intelligence. https://arxiv.org/pdf/1906.05833.pdf.

Landgrebe, J. and B. Smith (2021a, 16 November). An argument for the impossibility of machine intelligence. https://arxiv.org/abs/2111.07765.

Landgrebe, J. and B. Smith (2021b). Making AI meaningful again. *Synthese 198*, 2061–2081.

Landgrebe, J. and B. Smith (2023). *Why Machines Will Never Rule the World: Artificial Intelligence without Fear*. New York: Routledge.

Le, Q. V., M. Ranzato, et al. (2012). Building high-level features using large scale unsupervised learing. In *Proceedings of the 29th International Conference on Machine Learning*. Edinburgh, Scotland. https://icml.cc/2012/papers/73.pdf.

Lee, A. Y., J. Myers, and G. O. Rabin (2022). The structure of analog representation. *Noûs*. https://doi.org/10.1111/nous.12404.

Legg, S. and M. Hutter (2007). Universal intelligence: A definition of machine intelligence. *Minds and Machines 17*, 391–444. https://link.springer.com/content/pdf/10.1007/s11023-007-9079-x.pdf.

Leibniz, G. W. (1714). The Principles of Philosophy Known as Monadology. Modern English version by Jonathan Bennett (July 2007), *Some Texts from Early Modern Philosophy*, http://www.earlymoderntexts.com/assets/pdfs/leibniz1714b.pdf.

Lethen, T. (2020). Kurt Gödel's anticipation of the Turing machine: A vitalistic approach. *History and Philosophy of Logic 41*(3), 252–264.

Levesque, H. J. (2017). *Common Sense, the Turing Test, and the Quest for Real AI*. Cambridge, MA: MIT Press.

Lewis, D. (1969, July). Lucas against mechanism. *Philosophy 44*(169), 231–233. See also Lewis 1979.

Lewis, D. (1979, September). Lucas against mechanism II. *Canadian Journal of Philosophy 9*(2), 373–376. See also Lewis 1969.

Lewontin, R. (2014, 8 May). The new synthetic biology: Who gains? *New York Review of Books 61*(8), 22–23. http://www.nybooks.com/articles/archives/2014/may/08/new-synthetic-biology-who-gains/.

Lipton, R. J. (2020, 19 November). Traveling salesman problem meets complexity theory. *Gödel's Lost Letter and P=NP*. https://rjlipton.wordpress.com/2020/11/19/traveling-salesman-problem-meets-complexity-theory/.

Locke, J. (1694). *An Essay concerning Human Understanding*. Oxford: Oxford University Press, 1975. Edited by Peter H. Nidditch.

Lu, T. K. and O. Purcell (2016, April). Machine life. *Scientific American 314*(4), 58–63.

Lucas, J. (1961, Apr.-Jul.). Minds, machines and Gödel. *Philosophy 36*(137), 112–127. http://users.ox.ac.uk/~jrlucas/Godel/mmg.html.

Lucas, J. (1968, January). Satan stultified: A rejoinder to Paul Benacerraf. *The Monist 52*(1), 145–158. See also Benacerraf 1967.

MacFarquhar, L. (2007, 12 February). Two heads: A marriage devoted to the mind-body problem. *The New Yorker*, 58–69. https://www.newyorker.com/magazine/2007/02/12/two-heads.

Marcus, G. (2023, 18 February). David beats go-liath. *The Road to AI We Can Trust*. https://garymarcus.substack.com/p/david-beats-go-liath.

Marcus, G., F. Rossi, and M. Veloso (Eds.) (2016). *Beyond the Turing Test*. Special issue of *AI Magazine* 37(1) (Spring). https://aaai.org/ojs/index.php/aimagazine/issue/view/213.

Markoff, J. (2012, 25 June). How many computers to identify a cat? 16,000. *New York Times*. https://www.nytimes.com/2012/06/26/technology/in-a-big-network-of-computers-evidence-of-machine-learning.html.

Markoff, J. (2020, 21 May). A case for cooperation between machines and humans. *New York Times*. https://www.nytimes.com/2020/05/21/technology/ben-shneiderman-automation-humans.html.

Martins, J. P. and S. C. Shapiro (1988). A model for belief revision. *Artificial Intelligence 35*(1), 25–79. http://www.cse.buffalo.edu/~shapiro/Papers/marsha88.pdf.

McCarthy, J. (1979). Ascribing mental qualities to machines. In M. D. Ringle (Ed.), *Philosophical Perspectives in Artificial Intelligence*, pp. 161–195. Brighton, Sussex, UK: Harvester Press. http://jmc.stanford.edu/articles/ascribing.html.

McCarthy, J., M. Minsky, N. Rochester, and C. Shannon (1955). A proposal for the Dartmouth Summer Research Project on Artificial Intelligence. http://www-formal.stanford.edu/jmc/history/dartmouth.html, 31 August.

McCarthy, T. (2018). Normativity and mechanism. In G. Hellman and R. T. Cook (Eds.), *Hilary Putnam on Logic and Mathematics*, pp. 93–113. Cham, Switzerland: Springer Nature Switzerland.

McCulloch, W. S. and W. H. Pitts (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics 7*, 114–133. http://www.cs.cmu.edu/~epxing/Class/10715/reading/McCulloch.and.Pitts.pdf. Reprinted in Warren S. McCulloch, *Embodiments of Mind* (Cambridge, MA: MIT Press, 1965): 19–39.

McMillan, R. (2013, 7 July). The end of digital tyranny: Why the future of computing is analog. *Wired*. http://www.wired.com/wiredenterprise/2013/07/analogfuture/.

Mendelsohn, D. (2015, 4 June). The robots are winning! *New York Review of Books 62*(10), 51–54. http://www.nybooks.com/articles/archives/2015/jun/04/robots-are-winning/.

Mendelson, E. (1990, May). Second thoughts about Church's thesis and mathematical proofs. *Journal of Philosophy 87*(5), 225–233.

Metz, C. (2019a, 16 August). A.I. is learning from humans. Many humans. *New York Times*. https://www.nytimes.com/2019/08/16/technology/ai-humans.html.

Metz, C. (2019b, 11 November). We teach A.I. systems everything, including our biases. *New York Times*. https://www.nytimes.com/2019/11/11/technology/artificial-intelligence-bias.html.

Metz, C. (2021, 15 March). Who is making sure the A.I. machines aren't racist? *New York Times*. https://www.nytimes.com/2021/03/15/technology/artificial-intelligence-google-bias.html.

Miłkowski, M. (2022). Turing's conceptual engineering. *Philosophies 7*(3:69). https://doi.org/10.3390/philosophies7030069.

Miller, G. A., E. Galanter, and K. H. Pribram (1960). *Plans and the Structure of Behavior*. New York: Henry Holt.

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

Minsky, M. (1968). *Semantic Information Processing*. Cambridge, MA: MIT Press.

Minsky, M. (2006). *The Emotion Machine: Commonsense Thinking, Artificial Intelligence, and the Future of the Human Mind*. New York: Pantheon.

Minsky, M. and S. Papert (1974). *Artificial Intelligence: Condon Lectures*. University of Oregon Press. https://web.media.mit.edu/~minsky/papers/PR1971.html.

Mitchell, M. (2021, 16 December). What does it mean for AI to understand? *Quanta Magazine*. https://www.quantamagazine.org/what-does-it-mean-for-ai-to-understand-20211216/.

Mizoguchi, R. and Y. Kitamura (2009). A functional ontology of artifacts. *The Monist 92*(3), 387–402.

Monett, D., C. W. Lewis, and K. R. Thórisson (Eds.) (2020). *On Defining Artificial Intelligence*. Special Issue of *Journal of Artificial General Intelligence*, Vol. 11, No. 2 (February). https://content.sciendo.com/view/journals/jagi/11/2/jagi.11.issue-2.xml.

Monroe, C. R. and D. J. Wineland (2008, August). Quantum computing with ions. *Scientific American*, 64–71. http://www.cs.virginia.edu/~robins/Quantum_Computing_with_Ions.pdf.

Montague, R. (1960, December). Towards a general theory of computability. *Synthese 12*(4), 429–438.

Moor, J. H. (1976, October). An analysis of the Turing test. *Philosophical Studies 30*(4), 249–257.

Moor, J. H. (1978, September). Three myths of computer science. *British Journal for the Philosophy of Science 29*(3), 213–222.

Moor, J. H. (Ed.) (2003). *The Turing Test: The Elusive Standard of Artificial Intelligence*. Dordrecht, The Netherlands: Kluwer Academic.

Mycielski, J. (1983, February). The meaning of the conjecture $P \neq NP$ for mathematical logic. *American Mathematical Monthly 90*, 129–130.

Nagel, E., J. R. Newman, and D. R. Hofstadter (2001). *Gödel's Proof, Revised Edition*. New York: New York University Press.

Nagel, T. (1974, October). What is it like to be a bat? *Philosophical Review 83*(4), 435–450. https://www.sas.upenn.edu/~cavitch/pdf-library/Nagel_Bat.pdf.

Natarajan, P. (2017, 25 May). Calculating women. *New York Review of Books 64*(9), 38–39. http://www.nybooks.com/articles/2017/05/25/hidden-figures-calculating-women/.

Newell, A., J. Shaw, and H. A. Simon (1958). Elements of a theory of human problem solving. *Psychological Review 65*(3), 151–166.

Newell, A. and H. A. Simon (1976, March). Computer science as empirical inquiry: Symbols and search. *Communications of the ACM 19*(3), 113–126.

Newman, M. (1949, 25 June). A note on electronic automatic computing machines. *British Medical Journal 1*(4616), 1133.

Ohlsson, S., R. H. Sloan, G. Turàn, and A. Urasky (2015, 11 September). Measuring an Artificial Intelligence system's performance on a verbal IQ test for young children. http://arxiv.org/abs/1509.03390, http://arxiv.org/pdf/1509.03390v1.pdf.

Oommen, B. J. and L. G. Rueda (2005, May). A formal analysis of why heuristic functions work. *Artificial Intelligence 164*(1-2), 1–22.

Oppenheim, P. and H. Putnam (1958). Unity of science as a working hypothesis. *Minnesota Studies in the Philosophy of Science 2*(3), 3–35. https://philpapers.org/archive/OPPUOS.pdf.

Oppy, G. and D. Dowe (2020). The Turing test. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Fall 2020 ed.). Metaphysics Research Lab, Stanford University.

O'Regan, J. K. (2011). *Why Red Doesn't Sound Like a Bell: Understanding the Feeling of Consciousness*. New York: Oxford University Press.

Ovide, S. (2020, 25 November). We're still smarter than computers. For now. *New York Times On Tech*. https://www.nytimes.com/2020/11/25/technology/ai-gpt3.html.

Ovide, S. (2021, 15 March 2021). A.I. is not what you think. *New York Times On Tech*. https://www.nytimes.com/2021/03/15/technology/artificial-intelligence.html.

Peano, G. (1889). The principles of arithmetic, presented by a new method. In J. van Heijenoort (Ed.), *From Frege to Gödel: A Source Book in Mathematical Logic, 1879–1931*, pp. 83–97. Cambridge, MA: Harvard University Press. Trans. by Jean van Heijenoort.

Penrose, R. (1989). *The Emperor's New Mind: Concerning Computers, Minds, and the Laws of Physics*. Oxford: Oxford University Press.

Perlis, A. (1982, September). Epigrams on programming. *SIGPLAN Notices 17*(9), 7–13. http://pu.inf.uni-tuebingen.de/users/klaeren/epigrams.html.

Pettit, P. (1991, October). Realism and response-dependence. *Mind 100*(4), 587–626.

Petzold, C. (2008). *The Annotated Turing: A Guided Tour through Alan Turing's Historic Paper on Computability and the Turing Machine*. Indianapolis: Wiley.

Picard, R. (1997). *Affective Computing*. Cambridge, MA: MIT Press.

Piccinini, G. (2000). Turing's rules for the imitation game. *Minds and Machines 10*(4), 573–582.

Piccinini, G. (2004). The first computational theory of mind and brain: A close look at McCulloch and Pitts's "Logical calculus of ideas immanent in nervous activity". *Synthese 141*, 175–215. For a reply, see Aizawa 2010.

Piccinini, G. (2007, October). Computing mechanisms. *Philosophy of Science 74*(4), 501–526.

Piccinini, G. (2008). Computers. *Pacific Philosophical Quarterly 89*, 32–73. http://www.umsl.edu/~piccininig/Computers.pdf.

Piccinini, G. (2009). Computationalism in the philosophy of mind. *Philosophy Compass 4*(3), 515–532. 10.1111/j.1747-9991.2009.00215.x.

Piccinini, G. (2010, December). The resilience of computionalism. *Philosophy of Science 77*(5), 852–861.

Piccinini, G. (2011). The physical Church-Turing thesis: Modest or bold? *British Journal for the Philosophy of Science 62*, 733–769. http://www.umsl.edu/~piccininig/CT_Modest_or_Bold.pdf.

Piccinini, G. (2012). Computationalism. In E. Margolis, R. Samuels, and S. P. Stich (Eds.), *The Oxford Handbook of Philosophy of Cognitive Science*. Oxford University Press. http://www.umsl.edu/~piccininig/Computationalism.pdf.

Piccinini, G. (2015). *Physical Computation: A Mechanistic Account*. Oxford: Oxford University Press.

Piccinini, G. (2020). *Neurocognitive Mechanisms: Explaining Biological Cognition*. Oxford: Oxford University Press.

Piccinini, G. and N. G. Anderson (2020). Ontic pancomputationalism. In M. Cuffaro and S. Fletcher (Eds.), *Physical Perspectives on Computation, Computational Perspectives on Physics*. Cambridge, UK: Cambridge University Press. https://www.academia.edu/33271470/Ontic_Pancomputationalism.

Piccinini, G. and S. Bahar (2013). Neural computation and the computational theory of cognition. *Cognitive Science 34*, 453–488. http://www.umsl.edu/~piccininig/Neural_Computation_and_the_Computational_Theory_of_Cognition.pdf.

Piccinini, G. and C. Craver (2011). Integrating psychology and neuroscience: Functional analyses as mechanism sketches. *Synthese 183*(3), 283–311. http://philosophy.artsci.wustl.edu/files/philosophy/imce/integrating_psychology_and_neuroscience_functional_analyses_as_mechanism_sketches_0.pdf.

Pinker, S. and R. Jackendoff (2005). The faculty of language: What's special about it? *Cognition 95*, 201–236. http://pinker.wjh.harvard.edu/articles/papers/2005_03_Pinker_Jackendoff.pdf.

Piper, M. S. (2012). You can't eat causal cake with an abstract fork: An argument against computational theories of consciousness. *Journal of Consciousness Studies 19*(11–12), 154–190. https://www.academia.edu/10883735/You_Cant_Eat_Causal_Cake_with_an_Abstract_Fork_An_Argument_Against_Computational_Theories_of_Consciousness.

"PolR" (2009). An explanation of computation theory for lawyers. 11 November, http://www.groklaw.net/article.php?story=20091111151305785; also at http://www.groklaw.net/pdf2/ComputationalTheoryforLawyers.pdf.

Polya, G. (1957). *How to Solve It: A New Aspect of Mathematical Method, 2nd edition*. Garden City, NY: Doubleday Anchor.

Popper, K. (1953). *Conjectures and Refutations: The Growth of Scientific Knowledge*. New York: Harper & Row.

Popper, K. and J. C. Eccles (1977). *The Self and Its Brain: An Argument for Interactionism*. London: Routledge.

Pour-El, M. B. (1974, November). Abstract computability and its relation to the general purpose analog computer (some connections between logic, differential equations and analog computers). *Transactions of the American Mathematical Society 199*, 1–28.

Powers, R. (1988). *Prisoner's Dilemma*. New York: Harper Perrennial.

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

Proudfoot, D. (2013, July). Rethinking Turing's test. *Journal of Philosophy 110*(7), 391–411.

Proudfoot, D. (2017a). The Turing test—from every angle. In B. J. Copeland, J. P. Bowen, M. Sprevak, and R. Wilson (Eds.), *The Turing Guide*, pp. 287–300. Oxford: Oxford University Press.

Proudfoot, D. (2017b). Turing's concept of intelligence. In B. J. Copeland, J. P. Bowen, M. Sprevak, and R. Wilson (Eds.), *The Turing Guide*, pp. 301–307. Oxford: Oxford University Press.

Proudfoot, D. (2024). Wittgenstein and Turing on AI: Myth versus reality. In A. C. Helliwell, B. Ball, and A. Rossi (Eds.), *Wittgenstein and Artificial Intelligence; Vol. 1: Mind and Language*, pp. 17–37. London: Anthem Press. https://wpcpress.app.box.com/s/t10793t3yzx0f8qf4rwil1hjypb2csw6.

Putnam, H. (1960). Minds and machines. In S. Hook (Ed.), *Dimensions of Mind: A Symposium*, pp. 148–179. New York: New York University Press.

Putnam, H. (1965, March). Trial and error predicates and the solution to a problem of Mostowski. *Journal of Symbolic Logic 30*(1), 49–57. http://www.ninagierasimczuk.com/flt2013/wp-content/uploads/2013/01/Putnam_1965.pdf.

Pylyshyn, Z. W. (1984). *Computation and Cognition: Towards a Foundation for Cognitive Science*. Cambridge, MA: MIT Press. Ch. 3 ("The Relevance of Computation"), pp. 48–86, esp. §"The Role of Computer Implementation" (pp. 74–78).

Qian, L. and E. Winfree (2011, 3 June). Scaling up digital circuit computation with DNA strand displacement cascades. *Science 332*, 1196–1201. Reviewed in Reif 2011.

Quilty-Dunn, J., N. Porot, and E. Mandelbaum (2023). The best game in town: The re-emergence of the language of thought hypothesis across the cognitive sciences. *Behavioral and Brain Sciences*. https://doi.org/10.1017/S0140525X22002849.

Raatikainen, P. (2020). Gödel's incompleteness theorems. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Summer 2020 ed.). Metaphysics Research Lab, Stanford University.

Ramiro, C., M. Srinivasan, B. C. Malt, and Y. Xu (2018, 6 March). Algorithms in the historical emergence of word senses. *PNAS 115*(10), 2323–2328. https://doi.org/10.1073/pnas.1714730115.

Rapaport, W. J. (1986a, Summer). Philosophy, Artificial Intelligence, and the Chinese-room argument. *Abacus: The Magazine for the Computer Professional 3*, 6–17. Correspondence, *Abacus* 4 (Winter 1987): 6–7; 4 (Spring): 5–7; http://www.cse.buffalo.edu/~rapaport/Papers/abacus.pdf.

Rapaport, W. J. (1986b, June). Philosophy of Artificial Intelligence: A course outline. *Teaching Philosophy 9*(2), 103–120. http://www.cse.buffalo.edu/~rapaport/Papers/teachphil1986.pdf.

Rapaport, W. J. (1986c). Searle's experiments with thought. *Philosophy of Science 53*, 271–279. http://www.cse.buffalo.edu/~rapaport/Papers/philsci.pdf.

Rapaport, W. J. (1988a). Syntactic semantics: Foundations of computational natural-language understanding. In J. H. Fetzer (Ed.), *Aspects of Artificial Intelligence*, pp. 81–131. Dordrecht, The Netherlands: Kluwer Academic Publishers. http://www.cse.buffalo.edu/~rapaport/Papers/synsem.pdf; reprinted with numerous errors in Eric Dietrich (ed.) (1994), *Thinking Machines and Virtual Persons: Essays on the Intentionality of Machines* (San Diego: Academic Press): 225–273.

Rapaport, W. J. (1988b, December). To think or not to think: Review of Searle's *Minds, Brains and Science*. *Noûs 22*(4), 585–609. http://www.cse.buffalo.edu/~rapaport/Papers/2Tor-2T.pdf.

Rapaport, W. J. (1992a). Logic, predicate. In S. C. Shapiro (Ed.), *Encyclopedia of Artificial Intelligence, 2nd edition*, pp. 866–873. New York: John Wiley. http://www.cse.buffalo.edu/~rapaport/Papers/logic,predicate.pdf.

Rapaport, W. J. (1992b). Logic, propositional. In S. C. Shapiro (Ed.), *Encyclopedia of Artificial Intelligence, 2nd edition*, pp. 891–897. New York: John Wiley. http://www.cse.buffalo.edu/~rapaport/Papers/logic,propositional.pdf.

Rapaport, W. J. (1995). Understanding understanding: Syntactic semantics and computational cognition. In J. E. Tomberlin (Ed.), *AI, Connectionism, and Philosophical Psychology (Philosophical Perspectives, Vol. 9)*, pp. 49–88. Atascadero, CA: Ridgeview. http://www.cse.buffalo.edu/~rapaport/Papers/rapaport95-uu.pdf. Reprinted in Toribio, Josefa, & Clark, Andy (eds.) (1998), *Language and Meaning in Cognitive Science: Cognitive Issues and Semantic Theory (Artificial Intelligence and Cognitive Science: Conceptual Issues*, Vol. 4) (New York: Garland).

Rapaport, W. J. (1998). How minds can be computational systems. *Journal of Experimental and Theoretical Artificial Intelligence 10*, 403–419. http://www.cse.buffalo.edu/~rapaport/Papers/jetai-sspp98.pdf.

Rapaport, W. J. (1999). Implementation is semantic interpretation. *The Monist 82*, 109–130. http://www.cse.buffalo.edu/~rapaport/Papers/monist.pdf.

Rapaport, W. J. (2000a). Cognitive science. In A. Ralston, E. D. Reilly, and D. Hemmendinger (Eds.), *Encyclopedia of Computer Science, 4th edition*, pp. 227–233. New York: Grove's Dictionaries. http://www.cse.buffalo.edu/~rapaport/Papers/cogsci.pdf.

Rapaport, W. J. (2000b, October). How to pass a Turing test: Syntactic semantics, natural-language understanding, and first-person cognition. *Journal of Logic, Language, and Information 9*(4), 467–490. http://www.cse.buffalo.edu/~rapaport/Papers/TURING.pdf. Reprinted in James H. Moor, *The Turing Test: The Elusive Standard of Artificial Intelligence* (Dordrecht, The Netherlands: Kluwer Academic, 2003): 161–184.

Rapaport, W. J. (2002). Holism, conceptual-role semantics, and syntactic semantics. *Minds and Machines 12*(1), 3–59. http://www.cse.buffalo.edu/~rapaport/Papers/crs.pdf.

Rapaport, W. J. (2003). What did you mean by that? Misunderstanding, negotiation, and syntactic semantics. *Minds and Machines 13*(3), 397–427. http://www.cse.buffalo.edu/~rapaport/Papers/negotiation-mandm.pdf.

Rapaport, W. J. (2005, December). Implementation is semantic interpretation: Further thoughts. *Journal of Experimental and Theoretical Artificial Intelligence 17*(4), 385–417. https://www.cse.buffalo.edu//~rapaport/Papers/jetai05.pdf.

Rapaport, W. J. (2006a). How Helen Keller used syntactic semantics to escape from a Chinese room. *Minds and Machines 16*, 381–436. http://www.cse.buffalo.edu/~rapaport/Papers/helenkeller.pdf. See reply to comments, in Rapaport 2011.

Rapaport, W. J. (2006b, March). Review of Preston and Bishop 2002. *Australasian Journal of Philosophy 84*(1), 129–133. https://cse.buffalo.edu/~rapaport/Papers/crareview-ajp_129to133.pdf.

Rapaport, W. J. (2006c). The Turing test. In K. Brown (Ed.), *Encyclopedia of Language and Linguistics, 2nd Edition*, pp. Vol. 13, pp. 151–159. Oxford: Elsevier. http://www.cse.buffalo.edu/~rapaport/Papers/rapaport06-turingELL2.pdf.

Rapaport, W. J. (2011, Spring). Yes, she was! Reply to Ford's "Helen Keller was never in a Chinese room". *Minds and Machines 21*(1), 3–17. http://www.cse.buffalo.edu/~rapaport/Papers/Papers.by.Others/rapaport11-YesSheWas-MM.pdf.

Rapaport, W. J. (2012, January-June). Semiotic systems, computers, and the mind: How cognition could be computing. *International Journal of Signs and Semiotic Systems 2*(1), 32–71. https://tinyurl.com/rapaport12. Revised version published as Rapaport 2018a.

Rapaport, W. J. (2017a). On the relation of computing to the world. In T. M. Powers (Ed.), *Philosophy and Computing: Essays in Epistemology, Philosophy of Mind, Logic, and Ethics*, pp. 29–64. Cham, Switzerland: Springer. Preprint at http://www.cse.buffalo.edu/~rapaport/Papers/rapaport4IACAP.pdf.

Rapaport, W. J. (2017b, Fall). Semantics as syntax. *American Philosophical Association Newsletter on Philosophy and Computers 17*(1), 2–11. https://tinyurl.com/rapaport2016.

Rapaport, W. J. (2017c, Spring). What is computer science? *American Philosophical Association Newsletter on Philosophy and Computers 16*(2), 2–22. https://cdn.ymaws.com/www.apaonline.org/resource/collection/EADE8D52-8D02-4136-9A2A-729368501E43/ComputersV16n2.pdf.

Rapaport, W. J. (2018a). Syntactic semantics and the proper treatment of computationalism. In M. Danesi (Ed.), *Empirical Research on Semiotics and Visual Rhetoric*, pp. 128–176. Hershey, PA: IGI Global. References on pp. 273–307; http://www.cse.buffalo.edu/~rapaport/Papers/SynSemProperTrtmtCompnlism.pdf. Revised version of Rapaport 2012.

Rapaport, W. J. (2018b, Spring). What is a computer? A survey. *Minds and Machines 28*(3), 385–426. https://cse.buffalo.edu/~rapaport/Papers/rapaport2018-whatisacompr-MM.pdf.

Rapaport, W. J. (2020a). Syntax, semantics, and computer programs: Comments on Turner's *Computational Artifacts*. *Philosophy and Technology 33*, 309–321. https://link.springer.com/content/pdf/10.1007/s13347-019-00365-8.pdf.

Rapaport, W. J. (2020b). What is Artificial Intelligence? *Journal of Artificial General Intelligence 11*(2), 52–56. https://cse.buffalo.edu/~rapaport/Papers/rapaport2020whatisai.pdf.

Rapaport, W. J. (2023a). *Philosophy of Computer Science: An Introduction to the Issues and the Literature*. Hoboken, NJ; Oxford, UK: Wiley-Blackwell.

Rapaport, W. J. (2023b). A role for qualia. https://doi.org/10.1142/S2705078522500084, preprint at https://cse.buffalo.edu/~rapaport/Papers/qualia.pdf.

Reif, J. H. (2011, 3 June). Scaling up DNA computation. *Science 332*, 1156–1157. Review of Qian and Winfree 2011.

Rescorla, M. (2013, December). Against structuralist theories of computational implementation. *British Journal for the Philosophy of Science 64*(4), 681–707. http://philosophy.ucsb.edu/docs/faculty/papers/against.pdf.

Rescorla, M. (2014). A theory of computational implementation. *Synthese 191*, 1277–1307. http://philosophy.ucsb.edu/docs/faculty/papers/implementationfinal.pdf.

Rey, G. (1986, September). What's really going on in Searle's 'Chinese room'. *Philosophical Studies 50*(2), 169–185.

Rogers, Jr., H. (1959). The present theory of Turing machine computability. *Journal of the Society for Industrial and Applied Mathematics 7*(1), 114–130.

Romanycia, M. H. and F. J. Pelletier (1985). What is a heuristic? *Computational Intelligence 1*(2), 47–58. http://www.sfu.ca/~jeffpell/papers/RomanyciaPelletierHeuristics85.pdf.

Rubinoff, M. (1953, October). Analogue vs. digital computers—a comparison. *Proceedings of the IRE 41*(10), 1254–1262.

Russell, S. J. and P. Norvig (2020). *Artificial Intelligence: A Modern Approach; Fourth Edition*. Upper Saddle River, NJ: Pearson Education.

Sablé-Meyer, M., J. Fagot, S. Caparos, T. van Kerkoerle, M. Amalric, and D. Stanislas (2021). Sensitivity to geometric shape regularity in humans and baboons: A putative signature of human singularity. *PNAS 118*(16). https://doi.org/10.1073/pnas.2023123118.

Samuel, A. L. (1953, October). Computing bit by bit, or digital computers made easy. *Proceedings of the IRE 41*(10), 1223–1230.

Satariano, A. (2021, 21 April). Europe proposes strict rules for artificial intelligence. *New York Times*. https://www.nytimes.com/2021/04/16/business/artificial-intelligence-regulation.html.

Savage, N. (2016, March). When computers stand in the schoolhouse door. *Communications of the ACM 59*(3), 19–21. http://delivery.acm.org/10.1145/2880000/2875029/p19-savage.pdf.

Schaeffer, J., N. Burch, Y. Björnsson, A. Kishimoto, M. Müller, R. Lake, P. Lu, and S. Sutphen (2007, 19 July). Checkers is solved. *Sciencexpress*. https://www.researchgate.net/publication/231216842_Checkers_Is_Solved.

Scheutz, M. (1998). Implementation: Computationalism's weak spot. *Conceptus Journal of Philosophy 31*(79), 229–239. http://hrilab.tufts.edu/publications/scheutz98conceptus.pdf.

Schneider, S. (2009). The paradox of fiction. *Internet Encyclopedia of Philosophy*. https://www.iep.utm.edu/fict-par/.

Schweizer, P. (1998, November). The truly total Turing test. *Minds and Machines 8*(2), 263–272. https://tinyurl.com/schweizer1998.

Seabrook, J. (2019, 14 October). The next word. *The New Yorker*, 52–63. https://tinyurl.com/seabrook2019 See also follow-up letters at https://tinyurl.com/seabrook2019letters.

Searle, J. R. (1980). Minds, brains, and programs. *Behavioral and Brain Sciences 3*, 417–457.

Searle, J. R. (1982, 29 April). The myth of the computer. *New York Review of Books*, 3–6. See correspondence, same journal, 24 June 1982, pp. 56–57.

Searle, J. R. (1984). *Minds, Brains and Science*. Cambridge, MA: Harvard University Press.

Shagrir, O. (1997). Two dogmas of computationalism. *Minds and Machines 7*, 321–344. https://link.springer.com/content/pdf/10.1023/A:1008236522699.pdf.

Shagrir, O. (1999). What is computer science about? *The Monist 82*(1), 131–149.

Shagrir, O. (2006). Why we view the brain as a computer. *Synthese 153*, 393–416. https://oronshagrir.huji.ac.il/publications/why-we-view-brain-computer.

Shagrir, O. (2012, Summer). Computation, implementation, cognition. *Minds and Machines 22*(2), 137–148.

Shagrir, O. (2022). *The Nature of Physical Computation*. New York: Oxford University Press.

Shanahan, M., M. Crosby, B. Beyret, and L. Cheke (2020, November). Artificial Intelligence and the common sense of animals. *Trends in Cognitive Sciences 24*(11), 862–872. https://www.cell.com/action/showPdf?pii=S1364-6613%2820%2930216-3.

Shanker, S. (1987). Wittgenstein versus Turing on the nature of Church's thesis. *Notre Dame Journal of Formal Logic 28*(4), 615–649.

Shapiro, E. and Y. Benenson (2006, May). Bringing DNA computers to life. *Scientific American 294*(5), 44–51. http://www.wisdom.weizmann.ac.il/~udi/papers/ShapiroBenensonMay06.pdf.

Shapiro, S. (1993). Understanding Church's thesis, again. *Acta Analytica 11*, 59–77.

Shapiro, S. (2013). The open texture of computability. In B. J. Copeland, C. J. Posy, and O. Shagrir (Eds.), *Computability: Turing, Gödel, Church, and Beyond*, pp. 153–181. Cambridge, MA: MIT Press.

Shapiro, S. (2016). Idealization, mechanism, and knowability. In L. Horsten and P. Welch (Eds.), *Gödel's Disjunction: The Scope and Limits of Mathematical Knowledge*, pp. 189–208. Oxford: Oxford University Press.

Shapiro, S. C. (1992). Artificial Intelligence. In S. C. Shapiro (Ed.), *Encyclopedia of Artificial Intelligence, 2nd Edition*, pp. 54–57. New York: John Wiley & Sons. https://www.cse.buffalo.edu/~shapiro/Papers/ai.pdf. Revised version in Anthony Ralston, Edwin D. Reilly & David Hemmendinger (eds.), *Encyclopedia of Computer Science, 4th Edition* (New York: Grove's Dictionaries, 1993): 89–93, https://www.cse.buffalo.edu/~shapiro/Papers/ai-eofcs.pdf.

Shapiro, S. C. (1995, November). Computationalism. *Minds and Machines 5*(4), 517–524. http://www.cse.buffalo.edu/~shapiro/Papers/computationalism.pdf.

Shapiro, S. C. and J. P. Bona (2010). The GLAIR cognitive architecture. *International Journal of Machine Consciousness 2*(2), 307–332. https://www.cse.buffalo.edu/~shapiro/Papers/glairPublished.pdf.

Shapiro, S. C. and S. C. Kwasny (1975, August). Interactive consulting via natural language. *Communications of the ACM 18*(8), 459–462. http://www.cse.buffalo.edu/~shapiro/Papers/shakwa75.pdf.

Shapiro, S. C. and W. J. Rapaport (1991). Models and minds: Knowledge representation for natural-language competence. In R. Cummins and J. Pollock (Eds.), *Philosophy and AI: Essays at the Interface*, pp. 215–259. Cambridge, MA: MIT Press. http://www.cse.buffalo.edu/~rapaport/Papers/mandm.tr.pdf.

Shapiro, S. C. and M. Wand (1976, November). The relevance of relevance. Technical Report 46, Indiana University Computer Science Department, Bloomington, IN. https://legacy.cs.indiana.edu/ftp/techreports/TR46.pdf.

Shepherdson, J. and H. Sturgis (1963, April). Computability of recursive functions. *Journal of the ACM 10*(2), 217–255.

Shieber, S. M. (Ed.) (2004). *The Turing Test: Verbal Behavior as the Hallmark of Intelligence*. Cambridge, MA: MIT Press.

Shladover, S. E. (2016, June). What "self-driving" cars will really look like. *Scientific American*. http://endofdriving.org/wp-content/uploads/2016/10/What_Self-Driving_Cars_Will_Really_Look_Like-ShladoverScientificAmericanJune2016.pdf.

Sieg, W. (1994). Mechanical procedures and mathematical experience. In A. George (Ed.), *Mathematics and Mind*, pp. 71–117. New York: Oxford University Press. http://repository.cmu.edu/cgi/viewcontent.cgi?article=1248&context=philosophy.

Sieg, W. (1999, March). Hilbert's programs: 1917–1922. *Bulletin of Symbolic Logic 5*(1), 1–44. https://www.researchgate.net/publication/38373137_Hilbert%27s_Programs_1917-1922.

Sieg, W. (2000, 28 February). Calculations by man and machine: Conceptual analysis. Technical Report CMU-PHIL-104, Carnegie-Mellon University Department of Philosophy, Pittsburgh, PA. https://kilthub.cmu.edu/articles/journal_contribution/Calculations_by_Man_and_Machine_Conceptual_Analysis/6491045.

Sieg, W. (2007, June). On mind & Turing's machines. *Natural Computing 6*(2), 187–205. https://www.cmu.edu/dietrich/philosophy/docs/seig/OnmindTuringsMachines.pdf.

Sieg, W. (2008a). Church without dogma: Axioms for computability. In S. Cooper, B. Löwe, and A. Sorbi (Eds.), *New Computational Paradigms: Changing Conceptions of What Is Computable*, pp. 139–152. Springer. https://www.cmu.edu/dietrich/philosophy/docs/tech-reports/175_Sieg.pdf.

Sieg, W. (2008b). On computability. In A. Irvine (Ed.), *Philosophy of Mathematics*, pp. 525–621. Oxford: Elsevier. https://tinyurl.com/wsieg2008.

Sieg, W. (2020). Gödel's philosophical challenge (to Turing). In B. J. Copeland, C. J. Posy, and O. Shagrir (Eds.), *Computability: Turing, Gödel, Church, and Beyond*, pp. 183–202. Cambridge, MA: MIT Press. https://www.academia.edu/17065675/G%C3%B6del_s_philosophical_challenge_to_Turing_.

Silver, D. et al. (2018, 7 December). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science 362*(6419), 1140–1144.

Simon, H. A. (1959, June). Theories of decision-making in economics and behavioral science. *American Economic Review 49*(3), 253–283.

Simon, H. A. (1978). Rational decision-making in business organizations. In A. Lindbeck (Ed.), *Nobel Lectures, Economics 1969–1980*, pp. 343–371. Singapore: World Scientific, 1992. http://www.nobelprize.org/nobel_prizes/economic-sciences/laureates/1978/simon-lecture.pdf. Also, *American Economic Review* 69(4) (1979): 493–513.

Simon, H. A. (1995). Machine as mind. In K. M. Ford, C. Glymour, and P. J. Hayes (Eds.), *Android Epistemology*, pp. 23–40. Menlo Park, CA, and Cambridge, MA: AAAI Press/MIT Press.

Simon, H. A. (1996a). Computational theories of cognition. In W. O'Donohue and R. F. Kitchener (Eds.), *The Philosophy of Psychology*, pp. 160–172. London: SAGE Publications. Page references to reprint at https://tinyurl.com/simon1996.

Simon, H. A. (1996b). *The Sciences of the Artificial, Third Edition*. Cambridge, MA: MIT Press.

Simon, H. A. and A. Newell (1958, January-February). Heuristic problem solving: The next advance in operations research. *Operations Research 6*(1), 1–10.

Singer, N. and C. Metz (2019, 19 December). Many facial-recognition systems are biased, says U.S. study. *New York Times*. https://www.nytimes.com/2019/12/19/technology/facial-recognition-bias.html.

Sloman, A. (1984). The structure of the space of possible minds. In S. Torrance (Ed.), *The Mind and the Machine: Philosophical Aspects of Artificial Intelligence*, pp. 35–42. New York: John Wiley and Sons. http://www.cs.bham.ac.uk/research/projects/cogaff/81-95.html#49a.

Sloman, A. (1998, 25 January). Supervenience and implementation: Virtual and physical machines. http://www.cs.bham.ac.uk/research/projects/cogaff/Sloman.supervenience.and.implementation.pdf.

Sloman, A. (2020a, 30 January). How to trisect an angle (using P-Geometry). https://www.cs.bham.ac.uk/research/projects/cogaff/misc/trisect.html.

Sloman, A. (2020b). Varieties of evolved forms of consciousness, including mathematical consciousness. *Entropy 22*(6), 615ff. https://www.mdpi.com/1099-4300/22/6/615.

Sloman, A. and R. Chrisley (2003). Virtual machines and consciousness. In O. Holland (Ed.), *Machine Consciousness*, pp. 133–172. Charlottesville, VA: Imprint Academic. https://www.cs.bham.ac.uk/research/projects/cogaff/03.html#200302.

Sloman, A. and M. Croucher (1981). Why robots will have emotions. In *Proceedings of IJCAI 1981*, pp. 197–202. IJCAI. http://www.ijcai.org/Proceedings/81-1/Papers/039.pdf.

Smith, B. C. (2019). *The Promise of Artificial Intelligence: Reckoning and Judgment*. Cambridge, MA: MIT Press.

Smith, C. S. (2020, 2 January). Dealing with bias in artificial intelligence. *New York Times*. https://www.nytimes.com/2019/11/19/technology/artificial-intelligence-bias.html.

Soare, R. I. (2009). Turing oracle machines, online computing, and three displacements in computability theory. *Annals of Pure and Applied Logic 160*, 368–399. Preprint at http://www.people.cs.uchicago.edu/~soare/History/turing.pdf; published version at http://ac.els-cdn.com/S0168007209000128/1-s2.0-S0168007209000128-main.pdf?_tid=8258a7e2-01ef-11e4-9636-00000aab0f6b&acdnat=1404309072_f745d1632bb6fdd95f711397fda63ee2. A slightly different version appears as Soare 2013.

Soare, R. I. (2013). Interactive computing and relativized computability. In B. J. Copeland, C. J. Posy, and O. Shagrir (Eds.), *Computability: Turing, Gödel, Church, and Beyond*, pp. 203–260. Cambridge, MA: MIT Press. A slightly different version appeared as Soare 2009.

Soare, R. I. (2016). *Turing Computability: Theory and Applications*. Berlin: Springer.

Sprevak, M. (2017). Turing's model of the mind. In B. J. Copeland, J. P. Bowen, M. Sprevak, and R. Wilson (Eds.), *The Turing Guide*, pp. 277–285. Oxford: Oxford University Press. https://marksprevak.com/publications/turing-s-model-of-the-mind-2017/.

Sprevak, M. (2018). Triviality arguments about computational implementation. In M. Sprevak and C. Matteo (Eds.), *The Routledge Handbook of the Computational Mind*, pp. 175–191. London: Routledge. https://marksprevak.com/pdf/paper/Sprevak---Triviality%20arguments%20about%20implementation.pdf.

Sternberg, R. J. (1985). *Beyond IQ: A Triarchic Theory of Human Intelligence*. Cambridge, UK: Cambridge University Press.

Sterrett, S. G. (2020, December). The genius of the 'original imitation game' test. *Minds and Machines 30*(4), 469–486. https://link.springer.com/article/10.1007/s11023-020-09543-6.

Strawson, G. (2005). Intentionality and experience: Terminological preliminaries. In D. W. Smith and A. L. Thomasson (Eds.), *Phenomenology and Philosophy of Mind*, pp. 41–66. Oxford: Clarendon Press. https://www.academia.edu/397806/Intentionality_and_Experience_Terminological_Preliminaries.

Strawson, G. (2015, 27 February). The consciousness myth. *Times Literary Supplement* (5839), 14–15. Revised version at https://www.researchgate.net/publication/279326090_Consciousness_myth.

Suber, P. (1988). What is software? *Journal of Speculative Philosophy 2*(2), 89–119. Revised version at https://dash.harvard.edu/bitstream/handle/1/3715472/suber_software.html.

Sullivan, P. (2018, 9 February). A battle over diamonds: Made by nature or in a lab? *New York Times*. https://www.nytimes.com/2018/02/09/your-money/synthetic-diamond-jewelry.html.

Swade, D. D. (2017). Turing, Lovelace, and Babbage. In B. J. Copeland, J. P. Bowen, M. Sprevak, and R. Wilson (Eds.), *The Turing Guide*, pp. 249–262. Oxford: Oxford University Press.

Tedre, M. (2015). *The Science of Computing: Shaping a Discipline*. Boca Raton, FL: CRC Press/Taylor & Francis.

Thagard, P. (2006). *Hot Thought: Mechanisms and Applications of Emotional Cognition*. Cambridge, MA: MIT Press.

Thagard, P. (2007, January). Coherence, truth, and the development of scientific knowledge. *Philosophy of Science 74*, 28–47.

Thurner, S., R. Hanel, and P. Klimek (2018). *Introduction to the Theory of Complex Systems*. Oxford: Oxford University Press.

Turing, A. M. (1936). On computable numbers, with an application to the *Entscheidungsproblem*. *Proceedings of the London Mathematical Society, Ser. 2, Vol. 42*, 230–265. Reprinted with corrections in Davis 1965, pp. 116–154; https://tinyurl.com/turing36.

Turing, A. M. (1939). Systems of logic based on ordinals. *Proceedings of the London Mathematical Society S2-45*(1), 161–228. https://londmathsoc.onlinelibrary.wiley.com/doi/epdf/10.1112/plms/s2-45.1.161. Reprinted with commentary in Copeland 2004, Ch. 3.

Turing, A. M. (1947). Lecture to the London Mathematical Society on 20 February 1947. In B. J. Copeland (Ed.), *The Essential Turing*, pp. 378–394. Oxford: Oxford University Press (2004). Editorial commentary on pp. 362–377; online at https://www.academia.edu/34875977/Alan_Turing_LECTURE_TO_THE_LONDON_MATHEMATICAL_SOCIETY_1947_.

Turing, A. M. (1948). Intelligent machinery. In B. J. Copeland (Ed.), *The Essential Turing*, pp. 410–432. Oxford: Oxford University Press (2004). https://weightagnostic.github.io/papers/turing1948.pdf.

Turing, A. M. (1950, October). Computing machinery and intelligence. *Mind 59*(236), 433–460. https://academic.oup.com/mind/article/LIX/236/433/986238.

Turing, A. M. (1951a). Can digital computers think? In B. J. Copeland (Ed.), *The Essential Turing*, pp. 476–486. Oxford: Oxford University Press (2004). Also in Shieber 2004, Ch. 6; online at https://aperiodical.com/wp-content/uploads/2018/01/Turing-Can-Computers-Think.pdf.

Turing, A. M. (1951b, 1996). Intelligent machinery, a heretical theory. *Philosophia Mathematica 4*(3), 256–260. Reprinted in Copeland 2004, Ch. 12 and Shieber 2004, Ch. 5; online at http://viola.informatik.uni-bremen.de/typo/fileadmin/media/lernen/Turing-_Intelligent_Machinery.pdf.

Turing, A. M. (1952, 14 August). The chemical basis of morphogenesis. *Philosophical Transactions of the Royal Society of London, Series B, Biological Science 237*(641), 37–72.

Turing, A. M., R. Braithwaite, G. Jefferson, and M. Newman (1952). Can automatic calculating machines be said to think? In B. J. Copeland (Ed.), *The Essential Turing*, pp. 487–506. Oxford: Oxford University Press (2004). Also in Shieber 2004, Ch. 7.

Turner, R. (2018). *Computational Artifacts: Towards a Philosophy of Computer Science*. Berlin: Springer.

Tye, M. (2017). *Tense Bees and Shell-Shocked Crabs: Are Animals Conscious?* New York: Oxford University Press.

Tye, M. (2021). Qualia. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Fall 2021 ed.). Metaphysics Research Lab, Stanford University. https://plato.stanford.edu/archives/fall2021/entries/qualia/.

von Neumann, J. (1966). *Theory of Self-Reproducing Automata*. Urbana, IL: University of Illinois Press. Arthur W. Burks (ed.).

Walsh, T. (2014, November-December). Candy Crush's puzzling mathematics. *American Scientist 102*(6), 430–433. https://www.americanscientist.org/article/candy-crushs-puzzling-mathematics.

Wang, H. (1957, January). A variant to Turing's theory of computing machines. *Journal of the ACM 4*(1), 63–92.

Wang, P. (2019). On defining Artificial Intelligence. *Journal of Artificial General Intelligence 10*(2), 1–37. https://content.sciendo.com/view/journals/jagi/10/2/article-p1.xml.

Wang, P. (2020). On defining Artificial Intelligence—author's response to commentaries. *Journal of Artificial General Intelligence 11*(2), 73–86. https://content.sciendo.com/downloadpdf/journals/jagi/11/2/article-p1.xml.

Wegner, P. (1997, May). Why interaction is more powerful than algorithms. *Communications of the ACM 40*(5), 80–91. http://www.cs.brown.edu/people/pw/papers/ficacm.ps.

Weizenbaum, J. (1976). *Computer Power and Human Reason*. New York: W.H. Freeman.

Wheeler, M. (2020, December). Deceptive appearances: The Turing test, response-dependence, and intelligence as an emotional concept. *Minds and Machines 30*(4), 513–532. https://link.springer.com/article/10.1007/s11023-020-09533-8.

Wiener, N. (1960, 6 May). Some moral and technical consequences of automation. *Science 131*(3410), 1355–1358. https://nissenbaum.tech.cornell.edu/papers/Wiener.pdf.

Wittgenstein, L. (1933–1934). *The Blue and Brown Books*. Oxford: Basil Blackwell, 1964.

Wittgenstein, L. (1958). *Philosophical Investigations, Third Edition*. New York: Macmillan. Trans. by G.E.M. Anscombe.

Woods, W. A. (2010). The right tools: Reflections on computation and language. *Computational Linguistics 36*(4), 601–630. http://cognet.mit.edu/pdfviewer/journal/coli_a_00018.

Xu, Y., B. C. Malt, and M. Srinivasan (2017, August). Evolution of word meanings through metaphorical mapping: Systematicity over the past millennium. *Cognitive Psychology 96*, 41–53. https://doi.org/10.1016/j.cogpsych.2017.05.005.

Yampolskiy, R. V. (2012). Turing test as a defining feature of AI-completeness. In X.-S. Yang (Ed.), *Artificial Intelligence, Evolutionary Computing and Metaheuristics*, pp. 3–17. Berlin: Springer. https://www.researchgate.net/publication/256169428_Turing_Test_as_a_Defining_Feature_of_AI-Completeness.

Yampolskiy, R. V. (2020, 11 July). Human $\neq$ AGI. https://arxiv.org/abs/2007.07710.

Zenil, H. and F. Hernández-Quiroz (2007). On the possible computational power of the human mind. In C. Gershenson, D. Aerts, and B. Edmonds (Eds.), *Worldviews, Science and Us: Philosophy and Complexity*, pp. 315–337. Singapore: World Scientific Publishing. Page references to preprint at http://arxiv.org/abs/cs/0605065.

Zimmer, C. (2021). *Life's Edge: The Search for What It Means to Be Alive*. New York: Dutton.

Zobrist, A. L. (2000). Computer games: Traditional. In A. Ralston, E. D. Reilly, and D. Hemmendinger (Eds.), *Encyclopedia of Computer Science, 4th edition*, pp. 364–368. New York: Grove's Dictionaries.