NEWSLETTER | The American Philosophical Association

Philosophy and Computers

SPRING 2017 VOLUME 16 | NUMBER 2

ISSN 2155-9708

APA NEWSLETTER ON

Philosophy and Computers

PETER BOLTUC, EDITOR

VOLUME 16 | NUMBER 2 | SPRING 2017

FROM THE EDITOR

Peter Boltuc

UNIVERSITY OF ILLINOIS, SPRINGFIELD

We are pleased to feature the article by Bill Rapaport, which was his John Barwise Prize acceptance speech at the 2016 Eastern Division meeting. Bill is a long-term friend of the Committee on Philosophy and Computers, and of this newsletter. In the spring of 2007, when I edited my first issue here and we were in a bit of a time-crunch, Bill took his important article, "Searle on Brains as Computers," which I believe may have been invited for a different venue, and gave it to us, quite selflessly. The present article, titled "What Is Computer Science?," does what's promised in the title—it is a thorough discussion of the main issues in computer science today (and in the recent past). The paper is based on the essential chapters of Bill's introductory work on philosophy of computer science, which has been taking shape on his website since at least 2004; yet, it goes beyond the introductory level and engages students and colleagues alike. The present article may serve as a model open source text to begin a class on any aspect of philosophical and general theoretical issues in computer science. (In the next issue of this newsletter we are going to have Bill's vital article "Semantics as Syntax" and a conversation between Rapaport and Selmer Bringsjord.)

We follow up with a provocative article: "Why Think That the Brain Is Not a Computer?" Its author, Marcin Miłkowski, is known as one of the main defenders—alongside Gualtiero Piccinini—of what I would call *the modern-moderate* version of computationalism. The current article provides an opportunity for Miłkowski to zero in on the main objections to this view. The next paper is a heavy-metal presentation of "predictive coding" as a platform for testing competing hypotheses about functionalities of *consciousness embodied in both biological and artificial systems*. The article is based upon (and to some degree provides a follow-up on) Jun Tani's important book *Exploring Robotic Minds: Actions, Symbols, and Consciousness as Self-Organizing Dynamic Phenomena* (Oxford University Press, 2016). Jun Tani, one of the leaders in *synthetic neurorobotics*, co-authored (with Jeff White) a more theoretical article on consciousness for the previous issue of this newsletter. Noteworthy is the fact that, for the current paper, the order of the authors has been reversed—here we learn firsthand some of the essential models in neuro-robotics. The article may be placed on the border, or maybe in a demarcation zone, between science and philosophy (and on the science side at that!). Yet, in its later sections it uses models based largely on Husserl to come up with a broad definition of consciousness. We look forward to reading the third and final part of this scientific trilogy in the following issue of the newsletter.

The remaining two articles are shorter and to a much larger degree belong to the realm of philosophy the way most departments of philosophy view it. While Tani and White related primarily to Husserl, Richard Evans focuses on Kant's conception of Constituted Mental Activity. Evans argues that the "Kantian cognitive architecture is a rule-induction computational process." Under certain constraints "the process' internal activities count as cognitive activities." This paper provides a philosophical background for a constructivist model of artificial cognitive architectures developed in Evans' presentation "A Kantian Cognitive Architecture" at 2016 IACAP (to appear in *Philosophical Studies*). Don Perlis, in his thought provoking paper, "I Am, Therefore I Think," argues in favor of *reflexive-self formulation of mind and consciousness* that can be studied as an engineering problem. Some historical issues in computer science touched on in this article provide a nice way to come back to the topics discussed in Rapaport's opening article.

I want to thank Marcello Guarini, the chair of this committee, and all committee members (who are listed in Marcello's note, *From the Chair*) for their support. Special thanks go to Lucia Vazquez, Interim Dean of the College of Liberal Arts and Sciences at the University of Illinois at Springfield, for continuing the tradition and making my work as the editor of this newsletter possible.

FROM THE CHAIR

Marcello Guarini

UNIVERSITY OF WINDSOR, WESTERN ONTARIO

The winner of the 2015 Barwise prize was Dr. William Rapaport, and we are fortunate to have the text of his acceptance talk—delivered at the 2016 American Philosophical Association Eastern Division meeting—in this issue of our newsletter.

We are also in position to announce the winner of the 2016 Barwise prize: Dr. Edward Zalta. Dr. Zalta is a Senior Research Scholar at the Center for the Study of Language and Information, Stanford University. Zalta has not only made a series of very high quality contributions to computational

metaphysics, but he is also one of the founders and the principal editor of the *Stanford Encyclopedia of Philosophy*. His contributions to philosophy and computing are ongoing, and the community of scholars in this area continues to benefit from his work. Dr. Zalta was very pleased to hear of the award. Unbeknownst to the APA Committee on Philosophy and Computers who selected Zalta for the award, Ed knew John Barwise personally and feels especially honored to be receiving this award. Dr. Zalta has agreed to accept the 2016 Barwise Prize at the 2018 APA Eastern meeting. We hope to announce details in the next issue of the newsletter.

Thanks go out to all members of the APA committee on Philosophy and Computers for their deliberations over the Barwise Prize. A special thanks goes to Susan Schneider, who has completed her term on the committee. My gratitude also goes out to the continuing members of the committee, Colin Allen, William Barry, Fritz J. McDonald, Gary Mar, Dylan E. Wittkower, and Piotr Boltuc. Finally, a special welcome to Gualtiero Piccinini, who has recently joined the committee.

Readers of the newsletter are encouraged to contact any member of the committee if they are interested in proposing or collaborating on a symposium at the APA that engages any of the wide range of issues associated with philosophy and computing. We are happy to continue facilitating the presentation of high quality research in this area.

FEATURED ARTICLE

What Is Computer Science?

William J. Rapaport
UNIVERSITY AT BUFFALO, THE STATE UNIVERSITY OF NEW YORK

ABSTRACT
A survey of various proposed definitions of "computer science," arguing that it is a "portmanteau" scientific study of a family of topics surrounding both theoretical and practical computing. Its single most central question is What can be computed (and *how*)? Four other questions follow logically from that central one: What can be computed *efficiently*, and how? What can be computed *practically*, and how? What can be computed *physically*, and how? What *should* be computed, and how?

> The Holy Grail of computer science is to capture the messy complexity of the natural world and express it algorithmically.
>
> – Teresa Marrin Nakra[1]

1 PHILOSOPHY OF COMPUTER SCIENCE

In 2004, I created a course on the philosophy of computer science;[2] a draft of a textbook based on the course is available online.[3] The book is intended for readers who might know some philosophy but no computer science, those who might know some computer science but no philosophy, and even those who know little or nothing about both. So, we begin by asking what *philosophy* is (primarily aimed at the computer-science audience), and, in particular: What is "the philosophy of *X*"? (where *X* = things like: science, psychology, history, and, of course, computer science).

I take the focal question of the philosophy of computer science to be: *What is computer science?* To answer this, we need to consider a series of questions, each of which leads to another: Is computer science a science, a branch of engineering, some combination of them, or something else altogether? To answer these, we need to ask what science is and what engineering is.

Whether science or engineering, computer science is surely *scientific*, so we next ask what it is a (scientific) study *of*. *Computers*? If so, then what is a computer? Or is computer science a study of *computation*? If so, then what is computation? What is an algorithm?[4] Algorithms are said to be procedures, or recipes, so what is a procedure? What is a recipe? What is the Church-Turing Computability Thesis (that our intuitive notion of computation is completely captured by the formal notion of Turing-machine computation)?[5] What is "hypercomputation" (i.e., the claim that the intuitive notion of computation goes beyond Turing-machine computation)?

Computations are expressed in computer programs, which are executed by computers, so *what is a computer program*? Are computer programs "implementations" of algorithms? If so, then what is an implementation? What is the relation of programs and computation to the world?[6] Are programs (scientific) theories? What is the difference between software and hardware? Are programs copyrightable texts, or are they patentable machines? Ontologically, they seem to be both texts and machines, yet legally they cannot be both copyrightable and patentable.[7] Can computer programs be verified?[8]

We then turn to issues in the philosophy of AI, focusing on the Turing Test and the Chinese Room Argument.[9]

Finally, we consider two questions in computer ethics, which, when I created the course, were not much discussed, but are now at the forefront of computational ethical debates: (1) Should we trust decisions made by computers?[10]—a question made urgent by the advent of automated vehicles. And (2) should we build "intelligent" computers? Do we have moral obligations towards robots? Can or should they have moral obligations towards us?

And, along the way, we look at how philosophers reason and evaluate logical arguments.[11]

Although these questions arise naturally from our first question (What is computer science?), they do not exhaust the philosophy of computer science. Many topics are *not* covered: the nature of information, social and economic uses of computers, the Internet, etc. However, rather than aiming for universal coverage, I seek to provide a

foundation for further discussion: Neither the course nor the book is designed to answer all (or even any) of the philosophical questions that can be raised about the nature of computer science, computers, and computation. Rather, they provide background knowledge to "bring students up to speed" on the conversations about these issues, so that they can read the literature for themselves and perhaps become part of the conversations by contributing their own views. The present paper is a synopsis of *Philosophy of Computer Science* (Chapter 3), based on my Barwise Prize talk at the APA.[12]

2 PRELIMINARY QUESTIONS

However, before investigate what computer science is, it's worth asking some preliminary questions.

2.1 WHAT IS THE NAME OF THIS DISCIPLINE?

Should we call the discipline "computer science" (which seems to assume that it is the *science* of a certain kind of *machine*), or "computer *engineering*" (which seems to assume that it is *not* a science, but a branch of engineering), or "*computing* science" (which seems to assume that it is the science of what those machines do), or "informatics" (which suggests that it is a mathe*matic*al discipline concerned with *infor*mation)?

In this essay—but only for convenience—I call it "computer science." However, by doing so, I do not presuppose that it is the science of computers. Think of the subject as being called by a 15-letter word "computerscience" that may have as little to do with computers or science as "cattle" has to do with cats. Or, to save space and suppress presuppositions, just think of it as "CS."

2.2 WHY ASK WHAT CS IS?

There are both academic and philosophical motivations for trying to define CS.

2.2.1 ACADEMIC MOTIVATIONS

There is the *political* question of where to locate a CS department: In a college, faculty, or school of *(arts and) science*? Of *engineering*? Or in its *own* college, faculty, or school (perhaps of informatics, along with communications and library science)?

There is the pedagogical question of what to teach in an introductory course: Programming? Computer literacy? The mathematical theory of computation? Or an introduction to several different branches of CS, including, perhaps, some of its history?

And there is the *publicity* question: How should a CS department advertise itself so as to attract good students? How should the discipline advertise itself so as to encourage primary- or secondary-school students to consider it as something to study in college or to consider it as an occupation? How should it advertise itself so as to attract more women and minorities to the field? How should it advertise itself to the public at large, so that ordinary citizens might have a better understanding of what CS is?

Different motivations may yield different definitions.

2.2.2 PHILOSOPHICAL MOTIVATIONS

The *philosophical* question concerns what CS "really" is. Is it like some other academic discipline (mathematics, physics, engineering)? Or is it *sui generis*?

To illustrate this difference, consider two very different comments by two Turing-award–winning computer scientists:[13] Marvin Minsky, a founder of artificial intelligence, once said:

> Computer science has such *intimate relations* with so many other subjects that *it is hard to see it as a thing in itself.*[14]

On the other hand, Juris Hartmanis, a founder of computational complexity theory, has said:

> Computer science *differs* from the known sciences so deeply that it has to be viewed as *a new species among the sciences*.[15]

3 TWO KINDS OF DEFINITIONS

3.1 AN EXTENSIONAL DEFINITION OF CS

As with most non-mathematical concepts, there are probably no necessary and sufficient conditions for being CS. At best, the various branches of the discipline share only a family resemblance. If no intensional definition can be given in terms of necessary and sufficient conditions, perhaps an extensional one can: Perhaps CS is simply whatever computer scientists do: "Computing has no nature. It is what it is because people have made it so."[16]

So, what do computer scientists do? Ordered from the most to the least abstract, this might range from the abstract mathematical theories of computation, computational complexity, and program development; through software engineering, operating systems, and AI; to computer architecture, chip design, networks, and social uses of computers. But this is less than satisfactory as a *definition*.

3.2 INTENSIONAL DEFINITIONS

In the absence of necessary and sufficient conditions or an extensional definition, we can ask what the *methodology* of CS is: Is it a methodology used elsewhere? Or is it a new methodology? And then we can ask what its *object* of study is: Does it study something that other disciplines also study? Or does it study something new? And is its object of study unique to CS?

As for methodology, CS has been said to be:

- an *art form*
  (Knuth has said that programs can be beautiful[17]),
- an *art and science*
  ("Science is knowledge which we understand so well that we can teach it to a computer; and if we don't fully understand something, it is an art to deal with it. . . . [T]he process of going from an art to a science means that we learn how to automate something"[18]),

- a *liberal art*
  (along the lines of the classical liberal arts of logic, math, or astronomy[19]),
- a branch of *mathematics*,[20]
- a *natural science*,[21]
- an *empirical study* of the artificial,[22]
- a combination of *science and engineering*,[23]
- just *engineering*,[24]
- or—generically—a "study"

But a study of what? Here is an alphabetical list of some of the objects that it

"traffics" in (to use Barwise's term[25]): algorithms, automation, complexity, computers, information, intelligence, numbers (and other mathematical objects), problem solving, procedures, processes, programming, symbol strings.

It is now time to look at some answers to our title question in more detail.

4 CS IS THE SCIENCE OF COMPUTERS

Allen Newell, Alan Perlis, and Herbert Simon argued that CS is exactly what its name suggests:

> Wherever there are phenomena, there can be a science to describe and explain those phenomena. . . . There are computers. Ergo, computer science is the study of computers.[26]

This argument is actually missing a premise to the effect that the science of computers (which the first two premises imply the existence of) is CS and not some other discipline.

Loui has objected to the first premise, noting that there are toasters, but no *science* of toasters.[27] Another objection to the first premise, explicitly considered by Newell, Perlis, and Simon, is that science studies only *natural* phenomena, but that computers are *non*-natural *artifacts*. They replied that there are also sciences of artifacts. But one could respond in other ways: Where is the dividing line between nature and artifice, anyway? Are birds' nests artificial? As Mahoney observes, not only are artifacts part of nature, we use them to study nature; indeed, nature itself might be computational in nature (so to speak).[28]

Another objection that they consider is to the missing premise, that the science of computers is not CS but some other subject: electrical engineering, or math, or, perhaps, psychology. They reply that CS overlaps each of these, but that no single discipline subsumes *all* of CS. Of course, this reply assumes that CS itself *is* a cohesive whole, which the extensional characterization in §3.1 seems to belie.

One of my department's deans once suggested that CS would eventually dissolve: The computer engineers would rejoin the EE department, the complexity theorists would join the math department, my AI colleagues might go into psychology, I would go back into philosophy, and so on. (In much the same way, microscopy dissolved into microbiology, optical engineering, etc.[29]).

The most significant objection that they consider is that CS studies something besides computers, namely, algorithms. Their reply is also significant: They change their definition! They conclude that CS is the science of computers *and surrounding phenomena, including algorithms*.

5 CS STUDIES ALGORITHMS

Donald Knuth starts his definition, largely without any argument other than a recitation of its history, roughly where Newell, Perlis, and Simon end theirs: "[C]omputer science is . . . the study of *algorithms*."[30] He cites, approvingly, a statement by the computer scientist George E. Forsythe that the central question of CS is: What can be automated? (On that question, see §14.1.1.1, below.)

Knuth goes on to point out, however, that you need computers in order to properly study algorithms, because "human beings are not precise enough nor fast enough to carry out any but the simplest procedures."[31] Are computers really necessary? Do you *need* a compass and straightedge to study geometry? (Hilbert probably didn't think so.) Do you *need* a microscope to study biology? (Watson and Crick probably didn't think so.) On the other hand, "deep learning" algorithms do seem to need computers in order to determine if they will really do what they are intended to do, and do so in real time.[32]

(We'll return to this in §11.)

So, just as Newell, Perlis, and Simon said that CS is the study of computers *and related phenomena* ***such as algorithms***, Knuth says that it is the study of algorithms *and related phenomena* ***such as computers***! Stated a bit more bluntly, Newell, Perlis, and Simon's definition comes down to this: CS is the science of *computers and algorithms*. Knuth's definition comes down to this: CS is the study of *algorithms and computers*. Ignoring for now the subtle difference between "science" and "study," what we have here are extensionally equivalent, but intensionally distinct, definitions. Shades of the blind men and the elephant!

To be fair, however, some ten years later, Knuth backed off from the "related phenomena" definition, more emphatically defining CS as "primarily the study of algorithms," because he "think[s] of algorithms as encompassing the whole range of concepts dealing with well-defined processes, including the structure of data that is being acted upon as well as the structure of the sequence of operations being performed," preferring the name 'algorithmics' for the discipline.[33] He also suggested that what computer scientists have in common (and that differentiates them from people in other disciplines) is that they are all "algorithmic thinkers."[34] (We'll return to this notion in §13.4.)

6 CS STUDIES INFORMATION

Others say "A plague on both your houses": CS is *not* the study of computers *or* of algorithms, but of *information*:

Forsythe said that CS is "the art and science of representing and processing *information* and, in particular, processing information with the logical engines called automatic digital computers."[35] Peter J. Denning defined it as "the body of knowledge dealing with the design, analysis, implementation, efficiency, and application of processes that transform *information*."[36] Jon Barwise said that computers are best thought of as "information processors," rather than as numerical "calculators" or as "devices which traffic in formal strings . . . of meaningless symbols."[37] And Hartmanis and Lin define CS this way:

> What is the object of study [of CS and engineering]? For the physicist, the object of study may be an atom or a star. For the biologist, it may be a cell or a plant. But computer scientists and engineers focus on information, on the ways of representing and processing information, and on the machines and systems that perform these tasks.[38]

Presumably, those who study "the ways of representing and processing" are the scientists, and those who study "the machines and systems" are the engineers. And, of course, it is not just information that is studied; there are the usual "related phenomena": Computer science studies how to *represent* and (algorithmically) *process* information, as well as the *machines* and systems that do this.

Simon takes an interesting position on the importance of computers as information processors:[39] He discusses two "revolutions": The first was the Industrial Revolution, which "substitut[ed] . . . mechanical energy for the energy of man [sic] and animal." The second was (were?) the Information Revolution(s), beginning with "written language," then "the printed book," and now the computer. He then points out that "The computer is a device endowed with powers of utmost generality for processing symbols." So, *pace* Barwise, the computer is an information processor *because* information is encoded in symbols.

But here the crucial question is: What is information? The term "information" as many people use it informally has many meanings: It could refer to Claude Shannon's mathematical theory of information;[40] or to Fred Dretske's or Kenneth Sayre's philosophical theories of information;[41] or to several others.[42]

As I noted in §1, the philosophy of information is really a separate (albeit closely related!) topic from the philosophy of computer science. But, if "information" isn't intended to refer to some specific theory, then it seems to be merely a vague synonym for "data" (itself a vague term!). As Michael Rescorla observes, "Lacking clarification [of the term 'information'], the description [of "computation as 'information processing' "] is little more than an empty slogan."[43]

And Gualtiero Piccinini has made the stronger claim that computation is distinct from information processing in *any* sense of 'information'. He argues, e.g., that semantic information *requires* representation, but computation does *not*; so, computation is distinct from semantic information processing.[44]

7 CS *IS* A NATURAL SCIENCE (OF *PROCEDURES*)

Then there are those who agree that CS is a natural science, but not of computers, algorithms, *or* information: Stuart C. Shapiro agrees with Newell, Perlis, and Simon that CS is a science, but he differs on what it is a science of, siding more with Knuth, but not quite: "Computer Science is a *natural science* that studies *procedures*."[45] Procedures are not natural *objects*, but they are measurable natural *phenomena*, in the same way that events are not (natural) "objects" but are (natural) "phenomena." On this point, Denning cites examples of the "discovery" of "information processes in the deep structures of many fields": biology, quantum physics, economics, management science, and even the arts and humanities, concluding that "computing is now a natural science," not (or no longer?) "a science of the artificial."[46] So, potential objections that sciences only study natural phenomena are avoided.

For Shapiro, procedures include, but are not limited to, algorithms. Whereas algorithms are typically considered to be precise, to halt, and to produce correct solutions, the more general notion allows for variations on these themes: (1) Procedures (as opposed to algorithms) may be imprecise, such as in a recipe. Does CS really study things like recipes? According to Shapiro (personal communication), the answer is "yes": An education in CS should help you write a better cookbook, because it will help you understand the nature of procedures better![47] (2) Procedures need not halt: A procedure might go into an infinite loop either by accident or, more importantly, on purpose, as in an operating system or a program that computes the infinite decimal expansion of π. (3) Nor do they have to produce a correct solution: A chess procedure does not always play optimally.

And CS *is* a *science*, which, like any science, has both theoreticians (who study the limitations on, and kinds of, possible procedures) as well as experimentalists.

And, as Newell and Simon suggest in their discussion of empirical results (see §8, below), there are "fundamental principles" of CS as a science.[48] Newell and Simon cite two: (1) The Physical Symbol System Hypothesis (a theory about the nature of symbols in the context of computers) and (2) Heuristic Search (a problem-solving method). Shapiro cites two others: (1) the Computability Thesis and (2) the Boehm-Jacopini Theorem that codifies "structured programming."[49]

Moreover, Shapiro says that computer science is *not just* concerned with algorithms and procedures that manipulate abstract information, but also with procedures that are linked to sensors and effectors that allow computers to operate in the real world. Procedures are, or could be, carried out in the real world by physical agents, which could be biological, mechanical, electronic, etc. Where do computers come in? According to Shapiro, a computer is simply "a general-purpose procedure-following machine." (But does a computer "follow" a procedure, or merely "execute" it?)

Several pleas for elaboration can be urged on Shapiro: Does his view de-emphasize the role of computers in CS,

or is it merely a version of the "surrounding phenomena" viewpoint (as with Knuth's view that CS is the study of the phenomena surrounding *algorithms*)?[50] Does the emphasis on procedures (rather than algorithms) lead us into the fraught territory of "hypercomputation"?[51] (We'll return to procedures in §13.3.)

8 CS IS *NOT* A *NATURAL* SCIENCE

In 1967, Simon joined with Newell and Perlis to argue that CS was the natural science of (the phenomena surrounding) *computers*. Two years later, in his classic book *The Sciences of the Artificial*, he said that it was a natural science of *the artificial*: Natural science studies things in the world, but he was careful not to say that the "things" must be "natural"! "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."[52] Indeed, "The world we live in today is much more a[n] . . . artificial world than it is a natural world. Almost every element in our environment shows evidence of human artifice."[53] So, (natural) science can study artifacts; the "sciences of the artificial" are natural sciences.

And then, in a classic paper from 1976, Newell and Simon updated their earlier characterization. Instead of saying that CS is the *science* of (the phenomena surrounding) computers, they now said that it is the "*empirical*" "*study*" of those phenomena, "not just the hardware, but *the programmed, living machine.*"[54]

The reason that they say that CS is not a *science* (in the classic sense) is that it doesn't always strictly follow the scientific (or "experimental") method. E.g., often *one* experiment will suffice to answer a question in CS, whereas in other sciences, *numerous* experiments have to be run. However, CS, like science, is *empirical*—because programs running on computers are *experiments*, though not necessarily like experiments in other experimental sciences. In fact, one difference between CS and other experimental sciences is that, in CS, the chief objects of study (the computers and the programs) are not "black boxes."[55] Most natural phenomena are things whose internal workings we cannot see directly but must infer from experiments we perform on them. But we know exactly how and why computers and computer programs behave as they do (they are "glass boxes," so to speak), because *we* (not nature) designed and built them. So, we can understand them in a way that we cannot understand more "natural" things. (However, although this is the case for "classical" computer programs, it is not the case for artificial-neural-network programs: "A neural network, however, was a black box";[56] see the comments about Google Translate in §11, below.)

By "programmed, living machines," they meant computers that are actually running programs—*not just* the static machines sitting there waiting for someone to use them (computers without programs), *nor* the static programs just sitting there on a piece of paper waiting for someone to load them into the computer, nor the algorithms just sitting there in someone's mind waiting for someone to express them in a programming language—but *processes that are actually running on a computer*. A *program* might be a static piece of text or the static way that a computer is hardwired. A *process* is a dynamic entity—the program in the "process" of actually being executed by the computer.

However, to study "programmed living machines," we certainly do need to study the algorithms that they are executing. After all, we need to know what they are doing—i.e., it seems to be necessary to know what algorithm a computer is executing. On the other hand, in order to study an algorithm, it does not seem to be *necessary* to have a computer around that can execute it or to study the computer that is running it. It can be helpful and valuable to study the computer and to study the algorithm actually being run on the computer, but the mathematical study of algorithms and their computational complexity doesn't *need* the computer. That is, the algorithm can be studied as a mathematical object, using only mathematical techniques, without necessarily executing it. It may be very much more convenient, and even useful, to have a computer handy, as Knuth notes, but it does not seem to be necessary. If that's so, then it would seem that *algorithms* are really the essential object of study of CS: Both views require algorithms, but only one requires computers. (We'll see a counterargument in §11.)

9 CS IS ENGINEERING, *NOT* SCIENCE

The software engineer Frederick P. Brooks, Jr., says that CS isn't science—which he calls "analytic"—because, according to him, it is not concerned with the "discovery of facts and laws."[57] Instead, he argues that it is "an engineering discipline." Computer scientists are "concerned with *making things*": with physical tools such as computers and with abstract tools such as algorithms, programs, and software systems for others to use; the computer scientist is a *toolmaker*. Computer science, he says, is concerned with the usefulness and efficiency of the tools it makes; it is *not*, he says, concerned with newness for its own sake (as scientists are). So, "the discipline we call 'computer science'" is really the "synthetic"—i.e., the *engineering*—discipline that is concerned with computers.

Here is his argument:[58]

1. "[A] science is concerned with the *discovery* of facts and laws."
2. "[T]he scientist *builds in order to study*; the engineer *studies in order to build*.
3. The purpose of engineering is to build things.
4. Computer scientists "are concerned with *making things*, be they computers, algorithms, or software systems."
5. ∴ "the discipline we call 'computer science' is in fact not a science but a *synthetic*, an engineering, discipline."

Let's accept premise 1 for now; it seems reasonable enough.[59]

The point of the second premise is this: If a scientist's goal is to discover facts and laws—i.e., to study rather than to build—then anything built by the scientist is only built for that ultimate purpose. But building is the ultimate goal of engineering, and any studying (or discovery of facts and laws) that an engineer does along the way to building something is merely done for that ultimate purpose. For science, building is a side-effect of studying; for engineering, studying is a side-effect of building. Both scientists and engineers, according to Brooks, build and study, but each focuses more on one than the other. (Does this remind you of the algorithms-vs.-computers dispute in §§4–5?)

The second premise supports the third, which defines engineering as a discipline whose goal is to build things, i.e., a "synthetic"—as opposed to an "analytic"—discipline. "We speak of engineering as concerned with 'synthesis,' while science is concerned with 'analysis'."[60] "Where physical science is commonly regarded as an analytic discipline that aims to find laws that generate or explain observed phenomena, CS is predominantly (though not exclusively) synthetic, in that formalisms and algorithms are created in order to support specific desired behaviors."[61] As with his claim about the nature of science in the first premise, the accuracy of Brooks's notion of engineering is a topic for another day.[62] So, let's also assume the truth of the second and third premises for the sake of the argument.

Clearly, if the fourth premise is true, then the conclusion will follow validly (or, at least, it will follow that computer scientists belong on the engineering side of the science–engineering, or studying–building, spectrum). But is it really the case that computer scientists are (only? principally?) concerned with building or "making things"? And, if so, what kind of things?

Moreover, computer scientists *do* discover and analyze facts and laws: Consider the theories of computation and of computational complexity, and the "fundamental principles" cited at the end of §7, above. Computer scientists devise *theories* about how to build things, and they try to *understand* what they build. All of this seems to be more science than engineering.

Interestingly, Brooks seems to suggest that computer scientists *don't* build computers, even if that's what he says in the conclusion of his argument! He says that "Even when we build a computer the computer scientist designs only the abstract properties—its architecture and implementation. Electrical, mechanical, and refrigeration engineers design the realization."[63] I think this passage is a bit confused: Briefly, I think the "abstract properties" *are* the design *for* the realization; the engineers *build* the realization (they don't *design* it).[64] But it makes an interesting point: Brooks seems to be saying that computer scientists only design *abstractions*, whereas other (real?) engineers *implement them in reality.* This is reminiscent of the distinction between the relatively abstract *specifications* for an algorithm (which typically lack detail) and its relatively concrete (and highly detailed) implementation in a computer *program*. Brooks (following Zemanek[65]) calls CS "the engineering of abstract objects": If engineering is a discipline that builds, then what computer-science-qua-engineering builds is *implemented abstractions*.

10 SCIENCE *XOR* ENGINEERING?

So, is CS a science of some kind (natural or otherwise), or is it not a science at all, but some kind of engineering? Here, we would be wise to listen to two skeptics about the exclusivity of this choice:

> Let's remember that there is only one nature—the division into science and engineering, and subdivision into physics, chemistry, civil and electrical, is a human imposition, not a natural one. Indeed, the division is *a human failure*; it reflects *our limited capacity to comprehend the whole*. That failure impedes our progress; it builds walls just where the most interesting nuggets of knowledge may lie.[66]

> Debates about whether [CS is] science or engineering can . . . be counterproductive, since we clearly are *both, neither, and more*. . . .[67]

11 CS AS "BOTH"

Could CS be both science *and* engineering—perhaps the *science* of computation and the *engineering* of computers—i.e., the study of the "programmed living machine"?

It certainly makes no sense to have a computer without a program. It doesn't matter whether the program is *hardwired* (in the way that a Turing machine is); i.e., it doesn't matter whether the computer is a *special*-purpose machine that can only do one task. The program is not separable from the machine; it is built into its structure. And it doesn't matter whether the program is a piece of *software* (like a program inscribed on a universal Turing machine's tape)—i.e., it doesn't matter whether the computer is a *general*-purpose machine that can be loaded with different "apps" allowing the *same* machine to do many *different* things. It is simply the case that, *without a program, the computer wouldn't be able to* ***do anything.*** So, insofar as CS is about computers and hence is engineering, it must also be about computation and hence a science (at least, a mathematical science).

But it also makes little sense to have a program without a computer to run it on. Yes, you can study the program mathematically (e.g., try to verify it) or study its computational complexity.[68]

> The ascendancy of logical abstraction over concrete realization has ever since been a guiding principle in computer science, which has kept itself organizationally almost entirely separate from electrical engineering. The reason it has been able to do this is that computation is primarily a logical concept, and only secondarily an engineering one. To compute is to engage in formal reasoning, according to certain formal symbolic rules, and *it makes no logical difference how the formulas are physically represented, or how the logical transformations of them are physically realized*.[69]

But what good would it be (for that matter, what fun would it be!) to have, say, a program for passing the Turing test that never had an opportunity to pass it? Thus, *without a computer, the program wouldn't be able to actually do anything*. So, insofar as CS is about computation and hence is science, it should (must?) also be about computers and hence an engineering discipline.

So, *computers require programs* in order for the *computer* to do anything, and *programs require computers* in order for the *program* to actually be able to do anything. This is reminiscent of Kant's slogan that "Thoughts without content are empty, intuitions without concepts are blind. . . . The understanding can intuit nothing, the senses can think nothing. Only through their union can knowledge arise."[70] Similarly, we can say, "Computers without programs are empty; programs without computers are blind. Only through the union of a computer with a program can computational processing arise." A good example of this is the need for computers to test certain "deep learning" algorithms that Google used in their Translate software: Without enough computing power, there was no way to prove that their connectionist programs would work as advertised.[71] So, CS must be both a science (that studies algorithms) and an engineering discipline (that builds computers).

But we need not be concerned with these two fighting words, because, fortunately, there are two very convenient terms that encompass both: 'scientific' and 'STEM'. Surely, not only natural science, but also engineering, not to mention "artificial science," "empirical studies," and mathematics are all *scientific*. And, lately, NSF and the popular press have taken to referring to "STEM" disciplines—science, technology, engineering, and mathematics—precisely in order to have a single term to emphasize their similarities and interdependence, and to avoid having to try to spell out differences among them.[72]

So let's agree for the moment that CS might be *both* science *and* engineering. What about Freeman's other two options: *neither* and *more*?

12 CS AS "MORE"

12.1 CS IS A NEW KIND OF ENGINEERING

Michael Loui defines CS as "the theory, design, and analysis of algorithms for processing [i.e., for storing, transforming, retrieving, and transmitting] information, and the implementations of these algorithms in hardware and in software."[73] He argues that CS is "a new species of engineering."[74] He first argues that CS is an engineering discipline on the grounds that engineering (1) is concerned with what *can* exist (as opposed to what *does* exist), (2) "has a scientific basis," (3) is concerned with "design," (4) analyzes "trade-offs," and (5) has "heuristics and techniques." "Computer science has all the significant attributes of engineering"; therefore, CS is a branch of engineering.[75]

Let's consider each of these "significant attributes": First, his justification that CS is *not* "concerned with . . . what *does* exist" is related to the claim that CS is not a natural science, but a science of human-made artifacts. We have already considered two possible objections to this: First, insofar as procedures are natural entities, CS—as the study of procedures—*can* be considered a natural science. Second, insofar as some artifacts—such as bird's nests, beehives, etc.—are natural entities, studies of artifacts can be considered to be scientific.

Next, according to Loui, the "scientific basis" of CS is mathematics. The scientific basis of "traditional engineering disciplines such as mechanical engineering and electrical engineering" is physics. This is what makes it "new"; we'll come back to this.

According to Loui, engineers apply the principles of the scientific base of their engineering discipline to "design" a product: "[A] computer specialist applies the principles of computation to design a digital system or a program."[76] But not all computer scientists (or "specialists") design systems or programs; some do purely theoretical work. And, in any case, if the scientific basis of CS is mathematics, then why does Loui say that computer "specialists" apply "the principles of *computation*"? I would have expected him to say that they apply the principles of *mathematics*. Perhaps he sees "computation" as being a branch of mathematics. Or perhaps he doesn't think that the abstract mathematical theory of computation is part of CS, but that seems highly unlikely, especially in view of his definition of computer science as including the theory and analysis of algorithms. It's almost as if he sees computer *engineering* as standing to computer science in the same way that mechanical or electrical engineering stand to physics. But then it is not computer science that is a branch of engineering.

Let's turn briefly to trade-offs: "To implement algorithms efficiently, the designer of a computer system must continually evaluate trade-offs between resources" such as time vs. space, etc.[77] This is true, but doesn't support his argument as well as it might. For one thing, it is not only system designers who evaluate such trade-offs; so do theoretical computer scientists—witness the abstract mathematical theory of complexity. And, as noted above, not all computer scientists design such systems. So, at most, it is only those who do who are doing a kind of engineering.

Finally, as for heuristics, Loui seems to have in mind rough-and-ready "rules of thumb" rather than formally precise theories in the sense of Newell and Simon.[78] (See §14.1.3, below, for more on this kind of heuristics.) Insofar as engineers rely on such heuristics,[79] and insofar as some computer scientists also rely on them, then those computer scientists are doing something that engineers also do. But so do many other people: Writers surely rely on rule-of-thumb heuristics ("write simply and clearly"); does that make them engineers? This is probably his weakest premise.

The second part of Loui's argument is to show how CS is a "new" kind of engineering:[80]

1. "[E]ngineering disciplines have a scientific basis."

2. "The scientific fundamentals of computer science . . . are rooted . . . in mathematics."

3. "Computer science is therefore a *new* kind of engineering." (italics added) This argument can be made valid by adding two missing premises:

A. Mathematics is a branch of science.

B. No other branch of engineering has mathematics as its basis.

We can assume from his first argument that CS *is* a kind of engineering. So, from that and 1, we can infer that CS (as an engineering discipline) must have a scientific basis. We need premise A so that we can infer that the basis of CS (which, by 2, is mathematics) is indeed a scientific one. Then, from B, we can infer that CS must differ from all other branches of engineering. It is, thus, mathematical engineering.

However, despite these arguments, Loui also says this: "It is impossible to define a reasonable boundary between the disciplines of computer science and computer engineering. *They are the same discipline.*"[81] But doesn't that contradict the title of his essay ("Computer Science Is an Engineering Discipline")?

12.2 CS IS A NEW KIND OF SCIENCE

Recall that Hartmanis said that "computer science differs from the known sciences so deeply that it has to be viewed as a new species among the sciences."[82] First, Hartmanis comes down on the side of CS being a science: It is a "new species *among the sciences*." A chimpanzee is a different species from a tiger "among the animals," but they are both animals.

But what does it mean to be "a new species" of science? Both chimps and tigers are species of animals, and both lions and tigers are species within the genus *Panthera*. Is the relation of computer science to other sciences more like the relation of chimps to tigers (relatively distant) or lions to tigers (relatively close)? A clue comes in Hartmanis's next sentence:

> This view is justified by observing that theory and experiments in computer science play a different role and do not follow the classic pattern in physical sciences.[83]

This strongly suggests that CS is not a *physical* science (such as physics or biology), and Hartmanis confirms this suggestion on page 5: "computer science, *though not a physical science*, is indeed a science."[84] The non-physical sciences are typically taken to include at least the social sciences (such as psychology) and mathematics. So, it would seem that the relation of CS to other sciences is more like that of chimps to tigers: distantly related species of the same, high-level genus. And, moreover, it would seem to put computer science either in the same camp as (either) the so*cial sciences or mathematics, or else in a brand-new camp of its own, i.e.*, sui generis.

Hartmanis offers this definition of CS:

> At the same time, it is clear that *the objects of study in computer science are information and the machines and systems which process and transmit information*. From this alone, we can see that CS is concerned with the abstract subject of information, which gains reality only when it has a physical representation, and the man-made devices which process the representations of information. The goal of computer science is to endow these information processing devices with as much intelligent behavior as possible.[85]

Although it may be "clear" to Hartmanis that information (an "abstract subject") is (one of) the "objects of study in computer science," he does not share his reasons for that clarity. Since, as we have seen, others seem to disagree that CS is the study of information (e.g., it could be the study of computers or the study of algorithms), it seems a bit unfair for Hartmanis not to defend his view. But he cashes out this promissory note when he says that "what sets [CS] apart from the other sciences" is that it studies "processes [such as information processing] that are not directly governed by physical laws."[86] And why are they not so governed? Because "information and its transmission" are "abstract entities."[87] This makes computer science sound very much like mathematics. That is not unreasonable, given that it was this aspect of CS that led Hartmanis to his ground-breaking work on computational complexity, an almost purely mathematical area of CS.

But it's not just information that is the object of study; it's also information-processing machines, i.e., computers. Computers, however, don't deal directly with information, because information is abstract, i.e., non-physical. For one thing, this suggests that, insofar as CS is a new species of *non-physical* science, it is not a species of *social* science: Despite its name, the "social" sciences deal with pretty physical things: societies, people, speech, etc.

Hartmanis explicitly says that CS *is* a science and is *not* engineering, but his comments imply that it is both. I don't think he can have it both ways. This is remiscent of the dialogue between Newell, Perlis, and Simon on the one hand, and Knuth on the other. Both Loui and Hartmanis agree that computer science is a new kind of something or other; each claims that the scientific and mathematical aspects of it are central; and each claims that the engineering and machinery aspects of it are also central. But one *calls* it "science," while the other *calls* it "engineering." Again, it seems to be a matter of point of view.

A very similar argument (that does not give credit to Hartmanis!) that CS is a new kind of *science* can be found in Denning and Rosenbloom.[88] We'll look at some of what they have to say in §13.1.

13 CS AS "NEITHER"

And now for some things completely different . . .

13.1 CS HAS ITS OWN PARADIGM

Hartmanis argued that CS was *sui generis* ***among the sciences***. Denning and Peter A. Freeman offer a slightly stronger argument to the effect that CS is neither

science, engineering, nor math; rather CS has a "unique paradigm."[89]

But their position is somewhat muddied by their claim that "computing is a fourth great domain *of science* alongside the physical, life, and social sciences."[90] That implies that CS is a science, though of a different kind, as Hartmanis suggested.

It also leaves mathematics out of science! In a related article published three months earlier in the same journal, Denning and Paul S. Rosenboom assert without argument that "mathematics … has traditionally not been considered a science."[91] Denying that math is a science allows them to avoid considering CS as a *mathematical* science.[92]

In any case, to justify their conclusion that CS is truly *sui generis*, Denning and Freeman need to show that it is not a physical, life, or social science. Denning and Rosenbloom say that "none [of these] studies computation per se."[93] This is only half of what needs to be shown; it also needs to be shown that CS doesn't study physical, biological, or social entities. Obviously, it does study such things, though that is not its focus. As they admit, CS is "used extensively in all the domains";[94] i.e., computation is used by scientists in these domains as a tool.

So, what makes CS different? Denning and Freeman give a partial answer:

> The central focus of the computing paradigm can be summarized as information processes—natural or constructed processes that transform information. . . . [T]he computing paradigm . . . is distinctively different because of its central focus on information processes.[95]

This is only a partial answer, because it only discusses the *object of study* (which, as we saw in §6, is somewhat vague).

The rest of their answer is provided in a table showing the methodology of CS (Table 2, p. 29), which comes down to their version of "computational thinking."[96] We'll explore what that is in §13.4.

Denning and Freeman's version of it is close to what I will present as "synthetic" computational thinking in §14.1.1.1.

13.2 CS IS THE STUDY OF COMPLEXITY

It has been suggested that CS is the study of *complexity*—not just the mathematical subject of "computational complexity," but complexity in general and in all of nature. Ceruzzi ascribes this to Jerome Wiesner.[97] But all Wiesner says is that "Information processing systems are but one facet of . . . communication sciences . . . that is, the study of . . . the problems of organized complexity'."[98] But even if computer science is part of a larger discipline ("communication sciences"?) that studies complexity, it doesn't follow that CS itself *is* the study of complexity.

According to Ceruzzi, Edsgar Dijkstra also held this view: "programming, when stripped of all its circumstantial irrelevancies, boils down to no more and no less than very effective thinking so as to avoid unmastered complexity."[99] It is hierarchical structure that "offers a standard way to handle complexity":[100]

> [P]rograms are built from programs. . . . Programs are compilations in another sense as well. Even the smallest sub-program is also a compilation of sub-components. Programmers construct sub-programs by assembling into a coherent whole such discrete program elements as data, data structures, and algorithms. The "engineering" in software engineering involves knowing how to assemble these components to produce the desired behavior.[101]

The idea that a complex program is "just" a construction from simpler things, each of which—recursively—can be analyzed down to the primitive operations and data structures of one's programming system (for a Turing machine, these would be the operations of printing and moving, and data structures constructed from '0's and '1's) is, first, the underlying way in which complexity can be dealt with and, second, where engineering (considered as a form of construction) comes into the picture.

But, again, at most this makes the claim that *part* of computer science is the study of complexity. CS certainly offers many techniques for handling complexity: structured programming, abstraction, modularity, hierarchy, top-down design, stepwise refinement, object-oriented programming, recursion, etc. So, yes, CS is one way—perhaps even the best way—to *manage* (or *avoid*) complexity, not that it is *the study of* it. What's missing from Dijkstra's argument, in any case, is a premise to the effect that computer science is the study of programming, but Dijkstra doesn't say that, either in "EWD 512: Comments at a Symposium" (1975) or in "EWD 611: On the Fact that the Atlantic Ocean has Two Sides" (1976), the document that Ceruzzi says contains that premise.[102]

But Denning et al. point out that viewing "'computer science [as] the study of abstraction and the mastering of complexity' . . . also applies to physics, mathematics, or philosophy";[103] no doubt many other disciplines also study complexity. So *defining* CS the study of complexity doesn't seem to be right.

13.3 CS IS THE PHILOSOPHY*(!) OF PROCEDURES*

Could CS be the study of procedures, yet be a branch of *philosophy* instead of science? One major introductory CS text claims that CS is neither a science nor the study of computers.[104] Rather, it is what they call "procedural epistemology," which they define (italics added) as:

> *the study of the structure of knowledge* from an *imperative* point of view, as opposed to the more *declarative* point of view taken by classical mathematical subjects. Mathematics provides a framework for dealing precisely with notions of "what is." Computation provides a framework for dealing precisely with notions of "how to."

And, of course, epistemology is, after all, a branch of philosophy.

"How to" is certainly important, and interestingly distinct from "what is." But this distinction is hard to make precise. Many imperative statements can be converted to declarative ones; e.g., each "*p* :- *q*" rule of a Prolog program can be interpreted either procedurally ("to achieve *p*, execute *q*") or declaratively ("*p* if *q*").

Or consider Euclid's *Elements*; it was originally written in "how to" form: To construct an equilateral triangle using only compass and straightedge, follow this algorithm.[105] (Compare: To compute the value of this function *using only the operations of a Turing-machine*, follow this algorithm.)[106] But today it is expressed in "what is" form: The triangle that is constructed (using only compass and straightedge) by following that algorithm is equilateral: "When Hilbert gave a modern axiomatization of geometry at the beginning of the present century, he asserted the bald existence of the line. Euclid, however, also asserted that it can be constructed."[107] Note that the declarative version of a geometry theorem can be considered to be a formal proof of the correctness of the procedural version. This is closely related to the notion of program verification.

But even if procedural language can be intertranslated with declarative language, the two are distinct. And surely CS is concerned with procedures! There is a related issue in philosophy concerning the difference between knowing *that* something is the case (knowing that a declarative proposition is true) and knowing *how* to do something (knowing a procedure for doing it). This, in turn, may be related to Knuth's view of programming as teaching a computer (perhaps a form of knowing-that), to be contrasted with the view of a machine-learning algorithm that allows a computer to learn on its own by being trained. The former can easily gain declarative "knowledge" of what it is doing so that it can be programmed to explain what it is doing; the latter not so easily.

13.4 CS IS COMPUTATIONAL THINKING

A popular way to describe CS is as a "way of thinking," that "algorithmic thinking" (about anything!) is what makes CS unique:

> CS is the new "new math," and people are beginning to realize that CS, like math, is unique in the sense that many other disciplines will have to adopt that way of thinking. It offers a sort of conceptual framework for other disciplines, and that's fairly new. . . . Any student interested in science and technology needs to learn to think algorithmically. That's the next big thing.
>
> – Bernard Chazelle[108]

Jeannette Wing's notion of "computational thinking"[109] is thinking in such a way that a problem's solution "can effectively be carried out by an information-processing agent."[110] Here, it is important not to limit such "agents" to computers, but to include humans! It may offer compromises on several controversies: It avoids the procedural-declarative controversy, by including both concepts, as well as others. Her definition of CS as "the study of computation—what can be computed and how to compute it" is nice, too, because the first conjunct clearly includes the theory of computation and complexity theory ("can" can include "can in principle" as well as "can efficiently"), and the second conjunct can be interpreted to include both software programming as well as hardware engineering. "Study" is nice, too: It avoids the science-engineering controversy.

"[T]o think computationally [is] to use abstraction, modularity, hierarchy, and so forth in understanding and solving problems"[111]—indeed, computational thinking involves all of those methods cited in §13.2 for handling complexity! Five years before Perlis defined CS as the science of *computers*, he emphasized what is now called computational *thinking*:

> [T]he purpose of . . . [a] first course in programming . . . is not to teach people how to program a specific computer, nor is it to teach some new languages. *The purpose of a course in programming is to teach people how to construct and analyze processes*. . . .
>
> A course in programming . . . , if it is taught properly, is concerned with abstraction: the abstraction of constructing, analyzing, and describing processes. . . .
>
> This, to me, is the whole importance of a course in programming. It is a simulation. The point is not to teach the students how to use ALGOL, or how to program the 704. These are of little direct value. The point is to make the students construct complex processes out of simpler ones (and this is always present in programming) in the hope that the basic concepts and abilities will rub off. A properly designed programming course will develop these abilities better than any other course.[112]

Here is another characterization of CS, one that also characterizes computational thinking:

> Computer science is in significant measure all about analyzing problems, breaking them down into manageable parts, finding solutions, and integrating the results. The skills needed for this kind of thinking apply to more than computer programming. They offer a kind of disciplined mind-set that is applicable to a broad range of design and implementation problems. These skills are helpful in engineering, scientific research, business, and even politics![113] Even if a student does not go on to a career in computer science or a related subject, these skills are likely to prove useful in any endeavor in which analytical thinking is valuable.[114]

But Denning finds fault with the notion of "computational thinking," primarily on the grounds that it is too narrow:

> Computation is present in nature even when scientists are not observing it or thinking about it. Computation is more fundamental than computational thinking. For this reason alone, computational thinking seems like an inadequate characterization of computer science.[115]

Note that, by "computation," Denning means Turing-machine computation. For his arguments about why it is "present in nature," see the discussion in §7, above.[116]

13.5 CS IS AI

> [Computer science] is the science of how machines can be made to carry out *intellectual processes*.[117]
>
> The goal of computer science is to endow these information processing devices with as much *intelligent behavior* as possible.[118]
>
> Computational Intelligence *is* the manifest destiny of computer science, the goal, the destination, the final frontier.[119]

These aren't exactly definitions of CS, but they could be turned into ones: CS is the study of (choose one): (a) how to get computers to do what humans can do; (b) how to make computers (at least) as "intelligent" as humans; (c) how to understand (human) cognition computationally.

The history of computers supports this: It is a history that began with how to get machines to do *some* human thinking (certain mathematical calculations, in particular), then more and more. Indeed, the Turing machine, as a model of computation, was motivated by how *humans* compute: Turing analyzes how humans compute, and then designs a computer program that does the same thing.[120] But the branch of CS that analyzes how humans perform a task and then designs computer programs to do the same thing is AI. So, *the Turing machine was the first AI program!*

But, as I will suggest in §14.1, defining CS as AI is probably best understood as a special case of its fundamental task: determining what tasks are computable.

13.6 CS IS MAGIC

> Any sufficiently advanced technology is indistinguishable from magic.
>
> – Arthur C. Clarke[121]

Could it be that the advanced technology of CS is not only indistinguishable from magic, but really *is* magic? Not magic as in tricks, but magic as in Merlin or Harry Potter? As one CS student put it,

> Computer science is very empowering. It's kind of like knowing magic: you learn the right stuff and how to say it, and out comes an answer that solves a real problem. That's so cool.
>
> – Euakarn (Som) Liengtiraphan[122]

Brooks makes an even stronger claim than Clarke:

> The programmer, like the poet, works only slightly removed from pure thought-stuff. He [sic] builds castles in the air, creating by the exertion of the imagination. . . . Yet the program construct, unlike the poet's words [or the magician's spells?], is real in the sense that it moves and works, producing visible outputs separate from the construct itself. . . . ***The magic of myth and legend has come true in our time.*** *One types the correct incantation on a keyboard, and a display screen comes to life, showing things that never were nor could be.*[123]

Of course, the main difference between "the magic of myth and legend" and how computers work is that the former lacks (or at least fails to specify) any causal connection between incantation and result, whereas computation is quite clear about the connection: Recall our emphasis on algorithms (and see the discussion in §14.1.1.2, below).

What is "magic"? One anthropologist defines magic as the human "use of symbols to control forces in nature."[124] Clearly, programming involves exactly that kind of use of symbols.[125]

How is magic supposed to work? The anthropologist James G. Frazer "had suggested that primitive people imagine magical impulses traveling over distance through 'a kind of invisible ether'."[126] That sounds like a description of electromagnetic waves: Think of electrical currents running from a keyboard to a CPU, information traveling across the Internet, or text messaging.

According to another anthropologist, Bronisław Malinowski,

> The magical act involves three components: the formula, the rite, and the condition of the performer. The rite consists of three essential features: the dramatic expression of emotion through gesture and physical attitude, the use of objects and substances that are imbued with power by spoken words, and, most important, the words themselves.[127]

A "wizard," *gesturing* with a "wand," *performs* a "spell" consisting of a *formula* expressed in the *words* of an arcane language; the spell has real-world effects, *imbuing objects with power*.

Abstracting away from "the dramatic expression of emotion," use of a computer involves gestures, perhaps not with a wand, but with a mouse, a trackpad, or a touchscreen: The computer itself can be thought of as "imbued with power" when we issue, perhaps not a spell, but a command, either spoken or typed. And the words (of a programming language, or even English; think: Siri) used by the programmer or user are surely important, so the "rite" criterion is satisfied. Computer programs can be thought of as formulas, and only those programmers who know how to use appropriate programming languages, or those users who have accounts on a computer, might be considered to be in the right "condition."

> [A symbol] can take on the qualities of the thing it represents, and it can take the place of its referent; indeed, as is evident in religion and magic, *the symbol can become the thing it represents*, and in so doing, the symbol takes on the power of its referent.[128]

We see this happening in computers when we treat icons on a desktop (such icons are symbols) or the screen output of a WYSIWYG word processor (such as a page of a Microsoft Word document) as if they were the very things they represent. Perhaps more significantly, we see this in the case of those computer simulations in which the simulation of something really *is* that (kind of) thing: In online banking, the computational simulation of transferring funds between accounts *is* the transferring of funds; (simulated) signatures on online Word or PDF documents carry legal weight; in AI, computationally simulated cognition (arguably) *is* cognition.[129] And an NRC report talks about user interfaces as "illusions":[130]

> Unlike physical objects, the virtual objects created in software are not constrained to obey the laws of physics. . . . In the desktop metaphor, for example, the electronic version of file folders can expand, contract, or reorganize their contents on demand, quite unlike their physical counterparts.[131]

So, perhaps computers are not just metaphorically magic (as Arthur C. Clarke might have said); they *are* magic!

But, of course, the main difference between "the magic of myth and legend" and how computers work is that the former lacks (or at least fails to specify) any causal connection between incantation and result, whereas computation is quite clear about the connection: Recall our emphasis on algorithms (and see the discussion in §14.1.1.2, below).

14 SO, WHAT IS COMPUTER SCIENCE?

Our exploration of the various answers suggests that there is no simple, one-sentence answer to our question. Any attempt at one is no better than the celebrated descriptions of an elephant by the blind men: Many, if not most or all, such attempts wind up describing the entire subject, but focusing on only one aspect of it. Recall Newell, Perlis, and Simon's and Knuth's distinct but logically equivalent definitions.

CS is the scientific study of a family of topics surrounding both abstract (or theoretical) and concrete (or practical computing)—a "portmanteau" discipline.[132]

Charles Darwin said that "all true classification . . . [is] genealogical."[133] CS's genealogy involves two historical traditions: (1) the study of algorithms and the foundations of mathematics (from ancient Babylonian mathematics,[134] through Euclid's geometry, to inquiries into the nature of logic, leading ultimately to the Turing machine) and (2) the attempts to design and construct a calculating machine (from the Antikythera Mechanism of ancient Greece; through Pascal's and Leibniz's calculators and Babbage's machines; to the ENIAC, iPhone, and beyond). So, modern CS is the result of a marriage between (or merger of) the engineering problem of building better and better automatic calculating devices with the mathematical (hence, scientific) problem of understanding the nature of algorithmic computation. And that implies that modern CS, to the extent that it is a single discipline, has *both* engineering *and* science in its DNA. Hence its portmanteau nature.

The topics studied in contemporary CS roughly align along a spectrum ranging from the mathematical theory of computing, at one end, to the engineering of physical computers, at the other, as we saw in §3.2. Newell, Perlis, and Simon were looking at this spectrum from one end; Knuth was looking at it from the other end. The topics share a family resemblance (and perhaps nothing more than that, except for their underlying DNA), not only to each other, but also to other disciplines (including mathematics, electrical engineering, information theory, communication, etc.), and they overlap with issues discussed in the cognitive sciences, philosophy (including ethics), sociology, education, the arts, and business.

14.1 FIVE CENTRAL QUESTIONS OF CS

In this section, I want to suggest that there are five central questions of CS. The single most central question is:

1. **A. What *can* be computed?**

But to answer that, we also need to ask:

1. **B. *How* can it be computed?**

Several other questions follow logically from that central one:

2. **What can be computed *efficiently*, and how?**
3. **What can be computed *practically*, and how?**
4. **What can be computed *physically*, and how?**
5. **What *should* be computed, and how?**

Let's consider each of these in a bit more detail.

14.1.1 COMPUTABILITY

***14.1.1.1 What** Is Computable?* "What can be computed?" (or: "What is computable?") is the central question, because all other questions presuppose it. The fundamental task of any computer scientist—whether at the purely mathematical or theoretical end of the spectrum, or at the purely practical or engineering end—is to determine whether there is a computational solution to a given problem, and, if so, how to implement it. But those implementation questions are covered by the rest of the questions on our list, and only make sense after the first question has been answered. (Alternatively, they facilitate answering that first question; in any case, they serve the goal of answering it.)

Question 1 includes the question of computability vs. non-computability. It is the question that Church, Turing, Gödel, and others were originally concerned with—*Which*

mathematical functions are computable?—and whose answer has been given as the Church-Turing Computability Thesis: A *function* is computable if and only if it is computable by a Turing machine (or any formalism logically equivalent to a Turing machine, such as Church's lambda calculus or Gödel's general recursive functions). It is important to note that not all functions are computable. If they were, then computability would not be an interesting notion. (A standard example of a *non*-computable function is the Halting Problem.)

Various branches of CS are concerned with identifying which problems can be expressed by computable functions. So, a corollary of the Computability Thesis is that a *task* is computable if and only if it can be expressed as a computable function.

Here are some examples:

- Is cognition computable? The central question of AI is whether the functions that describe cognitive processes are computable. (This is one reason why I prefer to call AI "computational cognition."[135]) Given the advances that have been made in AI to date, it seems clear that at least some aspects of cognition are computable, so a slightly more precise question is: How much of cognition is computable?[136]

- Consider Shannon's 1950 paper on chess: The principal question is: Can we mathematically analyze chess? In particular, can we *computationally* analyze it (suggesting that computational analysis is a branch or kind of mathematical analysis)—i.e., can we analyze it procedurally? I.e., can we play chess rationally?

- Is the weather computable?[137]

- Is fingerprint identification computable?[138]

- Is final-exam-scheduling computable? Faculty members in my department recently debated whether it was possible to write a computer program that would schedule final exams with no time conflicts and in rooms that were of the proper size for the class. Some thought that this was a trivial problem; others thought that there was no such algorithm (on the (perhaps dubious!) grounds that no one in the university administration had ever been able to produce such a schedule); in fact, this problem is NP-complete.[139]

This aspect of question 1 is close to Forsythe's famous one:

> The question "What can be automated?" is one of the most inspiring philosophical and practical questions of contemporary civilization.[140]

Although similar in intent, Forsythe's question can be understood in a slightly different way: Presumably, a process can be automated—i.e., done automatically, by a machine, without human intervention—if it can be expressed as an algorithm. That is, computable implies automatable. But automatable does not imply computable: Witness the invention of the direct dialing system in telephony, which automated the task of the human operator. Yes, direct dialing is computable, but it wasn't a computer that did this automation.[141]

***14.1.1.2 How** Is It Computable?* The "how" question is also important: CS cannot be satisfied with a mere existence statement to the effect that a problem is computable; it also requires a constructive answer in the form of an algorithm that explicitly shows *how* it is computable.

In a Calvin and Hobbes cartoon,[142] Calvin discovers that if you input one thing (bread) into a toaster, a different thing (toast) is output. Hobbes wonders what happened to the input. It didn't disappear, of course, nor did it "magically" turn into the output:

> Everything going on in the software [of a computer] has to be physically supported by something going on in the hardware. Otherwise the computer couldn't do what it does from the software perspective—**it doesn't work by magic.** But usually we don't have to know how the hardware works—only the engineer and the repairman do. We can act as though the computer just carries out the software instructions, period. **For all we care, as long as it works, it might as well *be* magic.**[143]

Rather, the toaster *did something* to the bread (heated it). That intervening process is the analogue of an algorithm for the bread-to-toast function. Finding "intervening processes" requires algorithmic thinking, and results in algorithms that specify the transformational relations between input and output. (Where behaviorism focused only on inputs and outputs, cognitive psychology focused on the intervening algorithms.[144])

So, just as, for any *x*, there can be a philosophy of *x*, so we can ask, given some *x*, whether there is a computational theory of *x*. Finding a computational solution to a problem requires "computational thinking," i.e., algorithmic (or procedural) thinking (see §13.4, above).

Computational thinking includes what I call the four Great Insights of CS:[145]

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

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

3. The *structural* insight:
   Only 3 grammar rules are needed to combine actions
   (sequence, selection, repetition)

4. The "closure" insight: Nothing else is needed.
   (This is the import of the Church-Turing Computability Thesis.)[146]

Computational thinking involves both synthesis and analysis:

Synthesis: Given a problem *P*,

1. express *P* as a mathematical function *FP* (or a collection of interacting functions; i.e., give an input-output specification of *P*);

2. try to find an algorithm AF_P for computing *FP* (i.e., for transforming the input to the output; then try to find an efficient and practical version of AF_P);

3. implement AF_P on a physical computer.

Note the similarity of synthetic computational thinking to David Marr's analysis of information processing.[147]

Analysis:

> Given a real-world process *P* (physical, biological, psychological, social, economic, etc.), try to find a computational process *AP* that models (describes, simulates, explains, etc.) *P*.

Note that, once found, *AP* can be re-implemented; this is why computers can (be said to) think![148]

14.1.2 EFFICIENT COMPUTABILITY

Question 2 is the question studied by the branch of computer science known as computational complexity theory. Given an algorithm, one question is how much time it will take to be executed and how much space (memory) it will need. A more general question is this: Given the set of computable functions, which of them can be computed in, so to speak, less time than the age of the universe or less space than the size of the universe. The principal distinction is whether a function is in the class called P (in which case, it is "efficiently" computable) or in the class *NP* (in which case it is not efficiently *computable* but it *is* efficiently "verifiable"):[149]

> Even children can multiply two primes, but the reverse operation—splitting a large number into two primes—taxes even the most powerful computers. The numbers used in asymmetric encryption are typically hundreds of digits long. Finding the prime factors of such a large number is like trying to unmix the colors in a can of paint, . . . "Mixing paint is trivial. Separating paint isn't."[150]

Almost all practical algorithms are in *P*. By contrast, one important algorithm that is in *NP* is the Boolean Satisfiability Problem: Given a molecular proposition of propositional logic with n atomic propositions, under what assignment of truth-values to those atomic propositions is the molecular proposition true (or "satisfied")? Whether $P = NP$ is one of the major open questions in mathematics and CS; most computer scientists both hope and believe that $P = NP$.[151]

14.1.3 PRACTICAL COMPUTABILITY

Question 3 is considered both by complexity theorists as well as by more practically-oriented software engineers. Given a computable function in P (or, for that matter, in *NP*) what are some *practically* efficient methods of actually computing it? E.g., under certain circumstances, some sorting algorithms are more efficient in a practical sense (e.g., faster) than others. Even a computable function that is in *NP* might be practically computable in special cases. And some functions might only be practically computable "indirectly" via a "heuristic": A *heuristic for problem p* can be defined as an *algorithm* for some problem *p'*, where the solution to *p'* is "good enough" as a solution to *p*.[152] Being "good enough" is, of course, a subjective notion; Oommen and Rueda call the "good enough" solution "a *sub-optimal* solution that, hopefully, is arbitrarily close to the *optimal*."[153] The idea is related to Simon's notion of bounded rationality: We might not be able to solve a problem *p* because of limitations in space, time, or knowledge, but we might be able to solve a different problem *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.[154] A classic case of this is the Traveling Salesperson Problem, an *NP*-problem for which software like Google Maps solves special cases for us every day (even if their solutions are only "satisficing" ones[155]).

14.1.4 PHYSICAL COMPUTABILITY

But since the only (or the best) way to decide whether a computable function really does what it claims to do is to execute it on a computer, computers become an integral part of CS. Question 4 brings in both empirical (hence scientific) and engineering considerations. Even a practically efficient algorithm for computing some function might run up against physical limitations. Here is one example: Even if, eventually, computational linguists devise practically efficient algorithms for natural-language "competence" (understanding and generation,[156] it remains the case that humans have a finite life span, so the infinite capabilities of natural-language competence are not really required (a Turing machine isn't needed; a push-down automaton might suffice). This is also the question that issues in the design and construction of real computers ("computer engineering") are concerned with. And it is where investigations into alternative physical implementations of computing (quantum, optical, DNA, etc.) come in.

14.1.5 ETHICAL COMPUTABILITY

Bruce Arden, elaborating Forsythe's question, said that "the basic question [is] . . . what can *and should* be automated."[157] Question 5 brings in ethical considerations.[158] Actually, the question is slightly ambiguous. It could simply refer to questions of practical efficiency: Given a sorting problem, which sorting algorithm *should* be used; i.e., which one is the "best" or "most practical" or "most efficient" in the actual circumstances? But this sense of "should" does not really differentiate this question from question 3.

It is the *ethical* interpretation that makes this question interesting: Suppose that there is a practical and efficient algorithm for making certain decisions (e.g., as in the

case of autonomous vehicles). There is still the question of whether we *should* use those algorithms to actually make decisions for us. Or let us suppose that the goal of AI—a computational theory of cognition—is practically and efficiently computable by physically plausible computers. One can and should still raise the question whether such "artificial intelligences" *should* be created, and whether we (their creators) have any ethical or moral obligations towards them, and vice versa![159] And there is the question of implicit biases that might be (intentionally or unintentionally) built into some machine-learning algorithms.

14.2 WING'S FIVE QUESTIONS

It may prove useful to compare my five questions with Wing's "Five Deep Questions in Computing":[160]

1. $P = NP$?
2. What is computable?
3. What is intelligence?
4. What is information?
5. (How) can we build complex systems simply?

All but the last, it seems to me, concern scientific (abstract, mathematical) issues: If we consider Wing's second question to be the same as our central one, then her first question can be rephrased as our "What is efficiently computable?," and her third can be rephrased as "How much of (human) cognition is computable?" (a special case of our central question). Her fourth question can then be seen as asking an ontological question about the nature of what it is that is computed (an aspect of our central question): numbers (0s and 1s)? symbols ('0's and '1's)? information in some sense (and, if so, in which sense)?

Wing's last question is ambiguous between two readings of "build": On a software reading, it can be viewed in an abstract (scientific, mathematical) way as asking a question about the structural nature of software (the issues concerning the proper use of the "goto" statement [Dijkstra, 1968] and structural programming would fall under this category). As such, it concerns the grammar rules; it is then an aspect of our central question. But it can also be viewed on a hardware reading as asking an engineering question: How should we—literally—build computers?

Interpreted in this way, Wing's five questions can be boiled down to two:

- What is computation such that only some things can be computed? (And what can be computed (efficiently), and how?)
- (How) can we build physical devices to perform these computations?

The first is equivalent to our questions 1–3, the second to our question 4. And, in this case, we see once again the two parts of the discipline: the scientific (or mathematical, or abstract) and the engineering (or concrete).

It is interesting and important to note that none of Wing's questions correspond to the ethical question 5.

15 CONCLUSION

To sum up, computer science is the (scientific, or STEM) study of:

- what problems can be solved,
- what tasks can be accomplished, and
- what features of the world can be understood . . .

. . . *computationally*, i.e., using a language with only:

- 2 nouns ('0', '1'),
- 3 verbs ('move', 'print', 'halt'),
- 3 grammar rules (sequence, selection, repetition; or just recursion), and
- nothing else,

and then to provide algorithms to show how this can be done:

- efficiently,
- practically,
- physically, and
- ethically.

I said that our survey *suggests* that there is no simple, one-sentence answer to the question: What is computer science? My definition above is hardly a simple sentence.

But our opening quotation—from an interview with a computational musician—comes closer, so I will end where I began:

> The Holy Grail of computer science is *to capture the messy complexity of the natural world* and *express it algorithmically*.
>
> – Teresa Marrin Nakra[161]

NOTES

1. Quoted in J. Davidson, "Measure for Measure: Exploring the Mysteries of Conducting," *The New Yorker*, August 21, 2006, 66.
2. W. J. Rapaport, "Philosophy of Computer Science: An Introductory Course," Teaching Philosophy 28, no. 4 (2005): 319–41. See syllabus and supporting documents at http://www.cse.buffalo.edu/~rapaport/510.html
3. W. J. Rapaport, *Philosophy of Computer Science*, 2017, http://www.cse.buffalo.edu/~rapaport/Papers/phics.pdf
4. Ibid., Ch. 8, is a close, line-by-line reading of sections of A. M. Turing, "On Computable Numbers, with an Application to the *Entscheidungsproblem*," *Proceedings of the London Mathematical Society, Ser. 2*, 42 (1936): 230–65.

5. See R. I. Soare, "Turing Oracle Machines, Online Computing, and Three Displacements in Computability Theory," *Annals of Pure and Applied Logic* 160 (2009): 368–99, §12, on this name.

6. As discussed in B. C. Smith, "Limits of Correctness in Computers," *ACM SIGCAS Computers and Society* 14-15 (1985): 18–26; see also W. J. Rapaport, "On the Relation of Computing to the World," in *Philosophy and Computing: Essays in Epistemology, Philosophy of Mind, Logic, and Ethics*, ed. T. M. Powers, Springer, forthcoming).

7. A. Newell, "Response: The Models Are Broken, the Models Are Broken," *University of Pittsburgh Law Review* 47 (1985-1986): 1023–31.

8. J. H. Fetzer, "Program Verification: The Very Idea," *Communications of the ACM* 31, no. 9 (1988): 1048–63.

9. A. M. Turing, "Computing Machinery and Intelligence," Mind 59, no. 236 (1950): 433–60; J. R. Searle, "Minds, Brains, and Programs," *Behavioral and Brain Sciences* 3 (1980): 417–75.

10. J. H. Moor, "Are There Decisions Computers Should Never Make?" *Nature and System* 1 (1979): 217–29.

11. Computer Science Curricula 2013 covers precisely these sorts of argument-analysis techniques under the headings of Discrete Structures [DS]/Basic Logic, DS/Proof Techniques, Social Issues and Professional Practice [SP] (in general), and SP/Analytical Tools (in particular). Many other CS2013 topics also overlap those in the philosophy of computer science. See http://ai.stanford.edu/users/sahami/CS2013/

12. I am grateful to Thomas M. Powers and Richard M. Rubin for comments and discussion at the 2017 APA Eastern Division session where I presented the material in the present essay. And I am especially grateful to the American Philosophical Association Committee on Philosophy and Computers for this distinct honor, which recognizes "contributions to areas relevant to philosophy and computing" (http://www.apaonline.org/?barwise). Because of my philosophy interests in philosophy of mind, I was inspired—by Hofstatder's review of Aaron Sloman, *The Computer Revolution in Philosophy* (1978), which quoted Sloman (p. 5) to the effect that a philosopher of mind who knew no AI was like a philosopher of physics who knew no quantum mechanics—to study AI at SUNY Buffalo with Stuart C. Shapiro. This eventually led to a faculty appointment in computer science at Buffalo. Along the way, my philosophy colleagues and I at SUNY Fredonia published one of the first introductory logic textbooks to use a computational approach (Schagrin et al., *Logic: A Computer Approach*, 1985). At Buffalo, I was amazed to discover that my relatively arcane philosophy dissertation on Meinong was directly relevant to Shapiro's work in AI, providing an intensional semantics for his SNePS semantic-network processing system (Shapiro and Rapaport, "SNePS Considered as a Fully Intensional Propositional Semantic Network," 1987; Shapiro and Rapaport, "Models and Minds: Knowledge Representation for Natural-Language Competence," 1991). And then I realized that the discovery of quasi-indexicals ("he himself," "she herself," etc.) by my dissertation advisor, Hector-Neri Castañeda ("'He': A Study in the Logic of Self-Consciousness," 1966), could repair a "bug" in a knowledge-representation theory that Shapiro had developed with another convert to computer science (from psychology), Anthony S. Maida [Maida and Shapiro, "Intensional Concepts in Propositional Semantic Networks," 1982]. This work was itself debugged with the help of yet another convert (from English), my doctoral student Janyce M. Wiebe (Rapaport et al., "Quasi-Indexicals and Knowledge Reports," 1997]. My work with Shapiro and our SNePS Research Group at Buffalo enabled me to rebut Searle ("Minds, Brains, and Programs") using "syntactic semantics" (Rapaport, "Philosophy, Artificial Intelligence, and the Chinese-Room Argument," 1986; Rapaport, "Syntactic Semantics: Foundations of Computational Natural-Language Understanding," 1988; Rapaport, "Understanding Understanding: Syntactic Semantics and Computational Cognition," 1995; Rapaport, "How to Pass a Turing Test: Syntactic Semantics, Natural-Language Understanding, and First-Person Cognition," 2000; Rapaport, ""Semiotic Systems, Computers, and the Mind: How Cognition Could Be Computing," 2012). Both of these projects, as well as one of my early Meinong papers (Rapaport, "How to Make the World Fit Our Language: An Essay in Meinongian Semantics," 1981), led me, together with another doctoral student (Karen Ehrlich) and (later) a colleague from Buffalo's Department of Learning and Instruction (Michael W. Kibby), to develop a computational and pedagogical theory of vocabulary acquisition from context (Rapaport and Kibby, "Contextual Vocabulary Acquisition as Computational Philosophy and as Philosophical Computation," 2007; Rapaport and Kibby, "Contextual Vocabulary Acquisition: From Algorithm to Curriculum," 2014).

13. As cited in J. Gal-Ezer and D. Harel, "What (Else) Should CS Educators Know?" *Communications of the ACM* 41, no. 9 (1998): 79.

14. M. Minsky, "Computer Science and the Representation of Knowledge," in *The Computer Age: A Twenty Year View*, ed. L. Dertouzos and J. Moses (Cambridge, MA: The MIT Press, 1979), 392–421. My italics.

15. J. Hartmanis, "Some Observations about the Nature of Computer Science," in *Foundations of Software Technology and Theoretical Computer Science*, volume 761 of *Lecture Notes in Computer Science*, ed. R. Shyamasundar (Berlin/Heidelberg: Springer, 1993), 1; my italics. Cf. J. Hartmanis, "On Computational Complexity and the Nature of Computer Science," *ACM Computing Surveys* 27, no. 1 (1995): 10.

16. M. S. Mahoney, *Histories of Computing*, ed. Thomas Haigh (Cambridge, MA: Harvard University Press, 2011), 109.

17. D. E. Knuth, "Computer Programming as an Art," *Communications of the ACM* 17, no. 12 (1974a): 670.

18. Ibid., 668.

19. A. Perlis, "The Computer in the University," in *Management and the Computer of the Future*, ed. M. Greenberger (Cambridge, MA: The MIT Press, 1962), 210; S. Lindell, "Computer Science as a Liberal Art: The Convergence of Technology and Reason," talk presented at Haverford College, January 24, 2001, http://www.haverford.edu/cmsc/slindell/Presentations/Computer%20Science%20as%20a%20Liberal%20Art.pdf

20. E. W. Dijkstra, "Programming as a Discipline of Mathematical Nature," *American Mathematical Monthly* 81, no. 6 (1974): 608–12.

21. J. McCarthy, "A Basis for a Mathematical Theory of Computation," in *Computer Programming and Formal Systems*, ed. P. Braffort and D. Hirschberg (North-Holland, 1963), page references to PDF version at http://www-formal.stanford.edu/jmc/basis.html; A. Newell, A. J. Perlis, and H. A. Simon, "Computer Science," *Science* 157, no. 3795 (1967): 1373–74; S. C. Shapiro, "Computer Science: The Study of Procedures," *Technical report*, Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY, 2001, http://www.cse.buffalo.edu/~shapiro/Papers/whatiscs.pdf

22. H. A. Simon, *The Sciences of the Artificial, Third Edition* (Cambridge, MA: The MIT Press, 1996).

23. Hartmanis, "Some Observations about the Nature of Computer Science"; Hartmanis, "On Computational Complexity and the Nature of Computer Science; M. C. Loui, "Computer Science Is a New Engineering Discipline," *ACM Computing Surveys* 27, no. 1 (1995): 31–32.

24. F. P. Brooks, Jr., "The Computer Scientist as Toolsmith II," *Communications of the ACM* 39, no. 3 (1996): 61–68.

25. J. Barwise, "For Whom the Bell Rings and Cursor Blinks," *Notices of the American Mathematical Society* 36, no. 4 (1989): 386–88.

26. Newell et al., "Computer Science, 1373.

27. M. C. Loui, "Computer Science Is an Engineering Discipline," *Engineering Education* 78, no. 3 (1987): 175.

28. Mahoney, *Histories of Computing*, 159ff.

29. D. Boorstin, *The Discoverers* (New York: Random House, 1983), 376. See further discussion in Rapaport, *Philosophy of Computer Science*, §3.5.3.

30. D. E. Knuth, "Computer Science and Its Relation to Mathematics," *American Mathematical Monthly* 81, no. 4 (1974): 323.

31. Ibid.

32. G. Lewis-Kraus, "The Great A.I. Awakening," *New York Times Magazine*, December 14, 2016, http://www.nytimes.com/2016/12/14/magazine/the-great-ai-awakening.html

33. D. E. Knuth, "Algorithmic Thinking and Mathematical Thinking," *American Mathematical Monthly* 92, no. 3 (1985): 170–71.
34. Ibid., 172.
35. G. E. Forsythe, "A University's Educational Program in Computer Science," *Communications of the ACM* 10, no. 1 (1967): 3. My italics.
36. P. J. Denning, "What Is Computer Science?" *American Scientist* 73 (1985): 16. My italics.
37. Barwise, "For Whom the Bell Rings and Cursor Blinks," 386–87.
38. J. Hartmanis and H. Lin, eds. "What Is Computer Science and Engineering?" in *Computing the Future: A Broader Agenda for Computer Science and Engineering* (Washington, D.C.: National Academy Press, 1992), 164.
39. H. A. Simon, "What Computers Mean for Man and Society," *Science* 195, 4283 (1977): 1186.
40. C. E. Shannon, "A Mathematical Theory of Communication," *The Bell System Technical Journal* 27 (1948): 379–423, 623–56.
41. F. Dretske, *Knowledge and the Flow of Information* (Oxford: Blackwell, 1981); K. M. Sayre, "Intentionality and Information Processing: An Alternative Model for Cognitive Science," *Behavioral and Brain Sciences* 9, no. 1 (1986): 121–65.
42. For a survey, see G. Piccinini, G. *Physical Computation: A Mechanistic Account* (Oxford: Oxford University Press, 2015), Ch. 14.
43. M. Rescorla, "The Computational Theory of Mind," in *The Stanford Encyclopedia of Philosophy*, winter 2015 edition, ed. E. N. Zalta, http://plato.stanford.edu/archives/win2015/entries/computational-mind/, §6.1.
44. Piccinini,*Physical Computation*, Ch. 14, §3.
45. Shapiro, "Computer Science: The Study of Procedures." My italics.
46. P. J. Denning, "Computing Is a Natural Science," *Communications of the ACM* 50, no. 7 (2007): 13–18.
47. M. Sheraton, "The Elusive Art of Writing Precise Recipes," *The New York Times*, May 2, 1981 (http://www.nytimes.com/1981/05/02/style/de-gustibus-the-elusive-art-of-writing-precise-recipes.html), discusses the difficulties of writing recipes.
48. A. Newell and H. A. Simon, "Computer Science as Empirical Inquiry Symbols and Search," *Communications of the ACM* 19, no. 3 (1976): 113–26.
49. C. Böhm and G. Jacopini, "Flow Diagrams, Turing Machines and Languages with Only Two Formation Rules," *Communications of the ACM* 9, no. 5 (1966): 366–71.
50. Knowing Shapiro, I strongly suspect that it is the latter.
51. Rapaport,*Philosophy of Computer Science*, Ch. 11.
52. Simon,*The Sciences of the Artificial*, 1.
53. Ibid., 2.
54. Newell and Simon, "Computer Science as Empirical Inquiry Symbols and Search," 113, 114. My italics.
55. Ibid., 114.
56. Lewis-Kraus, "The Great A.I. Awakening," §4.
57. Brooks, "The Computer Scientist as Toolsmith II."
58. Ibid., 61–62.
59. I discuss this issue in more detail in Rapaport, *Philosophy of Computer Science*, Ch. 4.
60. Simon,*The Sciences of the Artificial*, 4.
61. J. Hendler, N. Shadbolt, W. Hall, T. Berners-Lee, and D. Weitzner, "Web Science: An Interdisciplinary Approach to Understanding the Web," *Communications of the ACM* 51, no. 7 (2008): 63.
62. Covered in Rapaport, *Philosophy of Computer Science*, Ch. 5.
63. Brooks, "The Computer Scientist as Toolsmith II," 62, col. 1.
64. W. J. Rapaport, "Implementation Is Semantic Interpretation," *The Monist* 82 (1999): 109–30; W. J. Rapaport, "Implemention Is Semantic Interpretation: Further Thoughts," *Journal of Experimental and Theoretical Artificial Intelligence* 17, no. 4 (2005): 385–417.
65. Following H. Zemanek, "Was ist informatik? (What Is Informatics?)," *Elektronische Rechenanlagen (Electronic Computing Systems)* 13, no. 4 (1971): 157–71.
66. W. Wulf, "Are We Scientists or Engineers?" *ACM Computing Surveys* 27, no. 1 (1995): 56. My italics.
67. P. A. Freeman, "Effective Computer Science," *ACM Computing Surveys* 27, no. 1 (1995): 27. My italics.
68. M. C. Loui, "Computational Complexity Theory," *ACM Computing Surveys* 28, no. 1 (1996): 47–49; S. Aaronson, "Why Philosophers Should Care about Computational Complexity," in *Computability: Turing, Gödel, Church, and Beyond*, ed. B. J. Copeland, C. J. Posy, and O. Shagrir (Cambridge, MA: The MIT Press, 2013), 261–327.
69. J. Robinson, "Logic, Computers, Turing, and von Neumann," In *Machine Intelligence 13: Machine Intelligence and Inductive Learning*, ed. K. Furukawa, D. Michie, and S. Muggleton (Oxford: Clarendon Press, 1994), 12. My italics.
70. I. Kant, *Critique of Pure Reason* (New York: St. Martin's Press, 1781/1787), 93 (A51/B75).
71. Lewis-Kraus, "The Great A.I. Awakening," §2.
72. Nothing should be read into the ordering of the terms in the acronym: The original acronym was the less mellifluous "SMET"! And educators, perhaps with a nod to Knuth's views, have been adding the arts, to create "STEAM" (http://stemtosteam.org/).
73. Loui, "Computer Science Is an Engineering Discipline," 176.
74. Loui, "Computer Science Is a New Engineering Discipline," 1.
75. Loui, "Computer Science Is an Engineering Discipline," 176.
76. Ibid.
77. Ibid., 177.
78. Newell and Simon, "Computer Science as Empirical Inquiry Symbols and Search."
79. B. V. Koen, "Toward a Definition of the Engineering Method," *European Journal of Engineering Education* 13, no. 3 (1988): 307–15.
80. Loui, "Computer Science Is a New Engineering Discipline," 31.
81. Loui, "Computer Science Is an Engineering Discipline," 178. My italics.
82. Hartmanis, "Some Observations about the Nature of Computer Science," 1.
83. Ibid.
84. My italics; cf. Hartmanis, "Some Observations about the Nature of Computer Science," 6; Hartmanis, "On Computational Complexity and the Nature of Computer Science," 11.
85. Hartmanis, "Some Observations about the Nature of Computer Science," 5; my italics. Cf. Hartmanis, "On Computational Complexity and the Nature of Computer Science," 10.
86. Hartmanis, "On Computational Complexity and the Nature of Computer Science," 10.
87. Ibid., 8.
88. P. J. Denning and P. S. Rosenbloom, "Computing: The Fourth Great Domain of Science," *Communications of the ACM* 52, no. (2009): 27–29.
89. P. J. Denning and P. A. Freeman, "Computing's Paradigm," *Communications of the ACM*, 52, no. 12 (2009): 28.
90. Ibid., 29. My italics.
91. Denning and Rosenbloom, "Computing: The Fourth Great Domain of Science," 28.
92. An option that we explore in Rapaport, *Philosophy of Computer Science*, Ch. 3, §3.10.2.

93. Denning and Rosenbloom, "Computing: The Fourth Great Domain of Science," 28.

94. Ibid.

95. Denning and Freeman, "Computing's Paradigm," 29–30.

96. Ibid., 30.

97. P. Ceruzzi, "Electronics Technology and Computer Science, 1940–1975: A Coevolution," *Annals of the History of Computing* 10, no. 4 (1988): 268–70; J. Wiesner, "Communication Sciences in a University Environment," *IBM Journal of Research and Development* 2, no. 4 (1958): 268–75.

98. Quoted in Ceruzzi, "Electronics Technology and Computer Science, 1940–1975: A Coevolution," 269.

99. E. W. Dijkstra, "EWD 512: Comments at a Symposium," in *Selected Writings on Computing: A Personal Perspective* (New York: Springer-Verlag, 1975), §4, p. 3.

100. L. Lamport, "How to Write a 21st Century Proof," *Journal of Fixed Point Theory and Applications* 11, no. 1 (2012): 16, http://research.microsoft.com/en-us/um/people/lamport/pubs/proof.pdf

101. P. Samuelson, R. Davis, M. D. Kapor, and J. Reichman, "A Manifesto Concerning the Legal Protection of Computer Programs," *Columbia Law Review* 94, no. 8 (1994): 2326–27.

102. Khalil and Levy, however, do make that claim. H. Khalil and L. S. Levy, "The Academic Image of Computer Science," *ACM SIGCSE Bulletin* 10, no. 2 (1978): 31–33.

103. P. J. Denning, D. E. Comer, D. Gries, M. C. Mulder, A. Tucker, A. J. Turner, and P. R. Young, "Computing as a Discipline," *Communications of the ACM* 32, no. 1 (1989): 11.

104. H. Abelson, G. J. Sussman, and J. Sussman, *Structure and Interpretation of Computer Programs* (Cambridge, MA: The MIT Press, 1996), "Preface to the First Edition."

105. G. Toussaint, "A New Look at Euclid's Second Proposition," *The Mathematical Intelligencer* 15, no. 3 (1993): 12–23. http://www.perseus.tufts.edu/hopper/text?doc=Perseus:text:1999.01.0086:book=1:type=Prop:number=1

106. For further discussion of "to accomplish goal G, do procedure P," see Rapaport, "On the Relation of Computing to the World."

107. N. D. Goodman, "Intensions, Church's Thesis, and the Formalization of Mathematics," *Notre Dame Journal of Formal Logic* 28, no. 4 (1987): §4.

108. Bernard Chazelle, interviewed in G. Anthes, "Computer Science Looks for a Remake," *Computerworld*, May 1, 2006, http://www.computerworld.com/s/article/110959/Computer_Science_Looks_for_a_Remake

109. J. M. Wing, "Computational Thinking," *Communications of the ACM* 49, no. 3 (2006): 33–35, echoing S. Papert, *Mindstorms: Children, Computers, and Powerful Ideas* (New York: Basic Books, 1980).

110. J. M. Wing, "Computational Thinking: What and Why?" *The Link (Carnegie-Mellon University)*, November 17, 2010, https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf. See also M. Guzdial, "A Definition of Computational Thinking from Jeannette Wing," *Computing Education Blog*, March 22, 2011, https://computinged.wordpress.com/2011/03/22/a-definition-of-computational-thinking-from-jeanette-wing/

111. J. Scott and A. Bundy, "Creating a New Generation of Computational Thinkers," *Communications of the ACM* 58, no. 12 (2015): 37.

112. Perlis, "The Computer in the University," 209–210. My italics.

113. And even the humanities (C. Ruff, "Computer Science, Meet Humanities: In New Majors, Opposites Attract," *Chronicle of Higher Education* 62, no. 21 [2016]: A19—WJR footnote).

114. V. G. Cerf, "Computer Science in the Curriculum," *Communications of the ACM* 59, no. 3 (2016): 7.

115. P. J. Denning, "Beyond Computational Thinking," *Communications of the ACM* 52, no. 6 (2009): 30.

116. For more on computational thinking, see the homepage for the Center for Computational Thinking, http://www.cs.cmu.edu/~CompThink/

117. McCarthy, "A Basis for a Mathematical Theory of Computation," 1. My italics.

118. Hartmanis, "Some Observations about the Nature of Computer Science," 5. My italics. Cf. "On Computational Complexity and the Nature of Computer Science," 10.

119. E. A. Feigenbaum, "Some Challenges and Grand Challenges for Computational Intelligence," *Journal of the ACM* 50, no. 1 (2003): 39. Also: "Understanding the activities of an animal or human mind in algorithmic terms seems to be about the greatest challenge offered to computer science by nature." J. Wiedermann, "Simulating the Mind: A Gauntlet Thrown to Computer Science," *ACM Computing Surveys* 31, 3es (1999): 1.

120. Turing, "On Computable Numbers, with an Application to the *Entscheidungsproblem*," §9.

121. http://en.wikipedia.org/wiki/Clarke's_three_laws

122. Euakarn (Som) Liengtiraphan, quoted in S. Hauser, "Computing and Connecting," *Rochester Review* 79, no. 3 (2017): 16.

123. F. P. Brooks, Jr., *The Mythical Man-Month* (Reading, MA: Addison-Wesley, 1975), 7–8. My emphases.

124. Phillip Stevens, Jr., "Magic," in *Encyclopedia of Cultural Anthropology*, ed. D. Levinson and M. Ember (New York: Henry Holt, 1996), 721.

125. Rapaport, "On the Relation of Computing to the World."

126. J. G. Frazer, *The Golden Bough: A Study in Magic and Religion, 3rd ed.* (London: Macmillan, 1911–1915), 722.

127. Stevens, "Magic," 722, citing Malinowski.

128. Ibid., 724. My italics.

129. For further discussion, see W. J. Rapaport, "Semiotic Systems, Computers, and the Mind: How Cognition Could Be Computing," *International Journal of Signs and Semiotic Systems* 2, no. 1 (2012): §8.

130. Cited by Samuelson et al., "A Manifesto Concerning the Legal Protection of Computer Programs," 2324, notes 44 and 46; 2325, note 47.

131. Ibid. 2334.

132. L. Carroll, *Through the Looking-Glass*, available at http://www.gutenberg.org/files/12/12-h/12-h.htm

133. C. Darwin, *The Origin of Species*. [1872] (New York: Signet Classics, 1958), Ch. 14, § "Classification," p. 437.

134. D. E. Knuth, "Ancient Babylonian Algorithms," *Communications of the ACM* 15, no. 7 (1972): 671–77.

135. W. J. Rapaport, "Understanding Understanding: Syntactic Semantics and Computational Cognition," in *Philosophical Perspectives, Vol. 9: AI, Connectionism, and Philosophical Psychology*, ed. J. E. Tomberlin (Atascadero, CA: Ridgeview Publishing, 1995), 49–88; W. J. Rapaport, "What Did You Mean By That? Misunderstanding, Negotiation, and Syntactic Semantics," *Minds and Machines* 13, no. 3 (2003): 397–427.

136. W. J. Rapaport, "Semiotic Systems, Computers, and the Mind: How Cognition Could Be Computing," *International Journal of Signs and Semiotic Systems* 2, no. 1 (2012): §2, pp. 34–35.

137. See B. Hayes, "Calculating the Weather," *American Scientist* 95, no. 3 (2007).

138. See S. N. Srihari, "Beyond C.S.I.: The Rise of Computational Forensics," *IEEE Spectrum*, 2010, http://spectrum.ieee.org/computing/software/beyond-csi-the-rise-of-computational-forensics

139. http://www.cs.toronto.edu/~bor/373s13/L14.pdf

140. G. E. Forsythe, "Computer Science and Education," *Information Processing 68: Proceedings of IFIP Congress 1968*, 1025.

141. "Strowger Switch," https://en.wikipedia.org/wiki/Strowger_switch

142. http://www.gocomics.com/calvinandhobbes/2016/03/09, originally published March 12, 1986.

143. R. Jackendoff, *A User's Guide to Thought and Meaning* (Oxford: Oxford University Press, 2012), 99. Original italics, my boldface.

144. G. A. Miller, E. Galanter, and K. H. Pribram, *Plans and the Structure of Behavior* (Henry Holt, New York, 1960).

145. http://www.cse.buffalo.edu/~rapaport/computation.html

146. The exact number of nouns, verbs, or grammar rules depends on the formalism. E.g., some presentations add "read" or "erase" 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 of expression.

147. D. Marr, *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information* (New York: W. H. Freeman, 1982). See discussion in Rapaport, "On the Relation of Computing to the World."

148. W. J. Rapaport, "How to Pass a Turing Test: Syntactic Semantics, Natural-Language Understanding, and First-Person Cognition," *Journal of Logic, Language, and Information* 9, no. 4 (2000): 467–90.

149. Technically, P is the class of functions computable in "Polynomial time," and NP is the class of functions computable in "Non-deterministic Polynomial time."

150. T. Folger, "The Quantum Hack," *Scientific American* 314, no. 2 (2016): 52.

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

152. W. J. Rapaport, "How Minds Can Be Computational Systems," *Journal of Experimental and Theoretical Artificial Intelligence* 10 (1998): 406.

153. B. J. Oommen and L. G. Rueda, "A Formal Analysis of Why Heuristic Functions Work," *Artificial Intelligence* 164, nos. 1-2 (2005): 1.

154. For more on heuristics, see M. H. Romanycia and F. J. Pelletier, "What Is a Heuristic?" *Computational Intelligence* 1, no. 2 (1985): 47–58; and S. J. Chow, "Many Meanings of 'Heuristic'," *British Journal for the Philosophy of Science* 66 (2015): 977–1016.

155. H. A. Simon, "Computational Theories of Cognition," in *The Philosophy of Psychology*, ed. W. O'Donohue and R. F. Kitchener (London: SAGE Publications, 1996), 160–72.

156. S. C. Shapiro, "The Cassie Projects: An Approach to Natural Language Competence," In *EPIA 89: 4th Portugese Conference on Artificial Intelligence Proceedings*, ed. J. Martins and E. Morgado (Berlin: Springer-Verlag, 1989), 362–80. Lecture Notes in Artificial Intelligence 390. http://www.cse.buffalo.edu/sneps/epia89.pdf; Shapiro and Rapaport, "SNePS Considered as a Fully Intensional Propositional Semantic Network."

157. B. W. Arden, ed. *What Can Be Automated? The Computer Science and Engineering Research Study (COSERS)* (Cambridge, MA: The MIT Press, 1980), 29. My italics.

158. M. Tedre, *The Science of Computing: Shaping a Discipline* (Boca Raton, FL: CRC Press/Taylor and Francis, 2015), 167–68.

159. M. Delvaux, "Draft Report with Recommendations to the Commission on Civil Law Rules on Robotics," European Parliament Committee on Legal Affairs, 2016, http://www.europarl.europa.eu/sides/getDoc.do?pubRef=-//EP//NONSGML%2BCOMPARL%2BPE-582.443%2B01%2BDOC%2BPDF%2BV0//EN

160. J. M. Wing, "Five Deep Questions in Computing," *Communications of the ACM* 51, no. 1 (2008): 58–60.

161. Teresa Marrin Nakra, quoted in Davidson, "Measure for Measure: Exploring the Mysteries of Conducting," 66. My italics.

REFERENCES

Aaronson, S. "Why Philosophers Should Care about Computational Complexity." In *Computability: Turing, Gödel, Church, and Beyond*, edited by B. J. Copeland, C. J. Posy, and O. Shagrir, 261–327. Cambridge, MA: The MIT Press, 2013.

Abelson, H., G. J. Sussman, and J. Sussman. *Structure and Interpretation of Computer Programs*. Cambridge, MA: The MIT Press, 1996.

Anthes, G. "Computer Science Looks for a Remake." *Computerworld*, May 1, 2006. http://www.computerworld.com/s/article/110959/Computer_Science_Looks_for_a_Remake

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

Barwise, J. "For Whom the Bell Rings and Cursor Blinks." *Notices of the American Mathematical Society* 36, no. 4 (1989): 386–88.

Böhm, C. and G. Jacopini. "Flow Diagrams, Turing Machines and Languages with Only Two Formation Rules." *Communications of the ACM* 9, no. 5 (1966): 366–71.

Boorstin, D. *The Discoverers*. New York: Random House, 1983. Ch. 49: "The Microscope of Nature."

Brooks, Jr., F. P. "The Computer Scientist as Toolsmith II." *Communications of the ACM* 39, no. 3 (1996): 61–68.

———. *The Mythical Man-Month*. Reading, MA: Addison-Wesley, 1975.

Carroll, L. *Through the Looking-Glass*. 1871. Available at http://www.gutenberg.org/files/12/12-h/12-h.htm

Castañeda, H.-N. "'He': A Study in the Logic of Self-Consciousness." Ratio 8 (1966): 130–57.

Cerf, V. G. "Computer Science in the Curriculum." *Communications of the ACM* 59, no. 3 (2016): 7.

Ceruzzi, P. "Electronics Technology and Computer Science, 1940–1975: A Coevolution." *Annals of the History of Computing* 10, no. 4 (1988): 257–75.

Chow, S. J. "Many Meanings of 'Heuristic'." *British Journal for the Philosophy of Science* 66 (2015): 977–1016.

Darwin, C. *The Origin of Species*. [1872]. New York: Signet Classics, 1958.

Davidson, J. "Measure for Measure: Exploring the Mysteries of Conducting." *The New Yorker*, August 21, 2006, 60–69.

Davis, R., P. Samuelson, M. Kapor, and J. Reichman. "A New View of Intellectual Property and Software." *Communications of the ACM* 39, no. 3 (1996): 21–30.

Delvaux, M. "Draft Report with Recommendations to the Commission on Civil Law Rules on Robotics." European Parliament Committee on Legal Affairs, 2016. http://www.europarl.europa.eu/sides/getDoc.do?pubRef=-//EP//NONSGML%2BCOMPARL%2BPE-582.443%2B01%2BDOC%2BPDF%2BV0//EN

Denning, P. J. "What Is Computer Science?" *American Scientist* 73 (1985): 16–19.

———. "Computing Is a Natural Science." *Communications of the ACM* 50, no. 7 (2007): 13–18.

———. "Beyond Computational Thinking." *Communications of the ACM* 52, no. 6 (2009): 28–30.

Denning, P. J., D. E. Comer, D. Gries, M. C. Mulder, A. Tucker, A. J. Turner, and P. R. Young. "Computing as a Discipline." *Communications of the ACM* 32, no. 1 (1989): 9–23.

Denning, P. J., and P. A. Freeman. "Computing's Paradigm." *Communications of the ACM*, 52, no. 12 (2009): 28–30.

Denning, P. J., and P. S. Rosenbloom. "Computing: The Fourth Great Domain of Science." *Communications of the ACM* 52, no. (2009): 27–29.

Dijkstra, E. W. "Go To Statement Considered Harmful." *Communications of the ACM* 11, no. 3 (1968): 147–48.

———. "Programming as a Discipline of Mathematical Nature." *American Mathematical Monthly* 81, no. 6 (1974): 608–12.

———. "EWD 512: Comments at a Symposium." In *Selected Writings on Computing: A Personal Perspective*, 161–64. New York: Springer-Verlag, 1975.

———. "EWD 611: On the Fact that the Atlantic Ocean has Two Sides." In *Selected Writings on Computing: A Personal Perspective*, 268–76. New York: Springer-Verlag, 1976.

Dretske, F. *Knowledge and the Flow of Information*. Oxford: Blackwell, 1981.

Feigenbaum, E. A. "Some Challenges and Grand Challenges for Computational Intelligence." *Journal of the ACM* 50, no. 1 (2003): 32–40.

Fetzer, J. H. "Program Verification: The Very Idea." *Communications of the ACM* 31, no. 9 (1988): 1048–63.

Folger, T. "The Quantum Hack." *Scientific American* 314, no. 2 (2016): 48–55.

Forsythe, G. E. "A University's Educational Program in Computer Science." *Communications of the ACM* 10, no. 1 (1967): 3–8.

———. "Computer Science and Education." *Information Processing 68: Proceedings of IFIP Congress 1968*, 1025–39.

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

Frazer, J. G. *The Golden Bough: A Study in Magic and Religion, 3rd ed*. London: Macmillan, 1911–1915.

Freeman, P. A. "Effective Computer Science." *ACM Computing Surveys* 27, no. 1 (1995): 27–29.

Gal-Ezer, J., and D. Harel. "What (Else) Should CS Educators Know?" *Communications of the ACM* 41, no. 9 (1998): 77–84.

Goodman, N. D. "Intensions, Church's Thesis, and the Formalization of Mathematics." *Notre Dame Journal of Formal Logic* 28, no. 4 (1987): 473–89.

Guzdial, M. "A Definition of Computational Thinking from Jeannette Wing." *Computing Education Blog*. March 22, 2011. https://computinged.wordpress.com/2011/03/22/a-definition-of-computational-thinking-from-jeanette-wing/

Hartmanis, J. "Some Observations about the Nature of Computer Science." In *Foundations of Software Technology and Theoretical Computer Science*, volume 761 of *Lecture Notes in Computer Science*, edited by R. Shyamasundar, 1–12. Berlin/Heidelberg: Springer, 1993.

———. "On Computational Complexity and the Nature of Computer Science." *ACM Computing Surveys* 27, no. 1 (1995): 7–16. Reprinted from *Communications of the ACM* 37, no. 10 (October 1994): 37–43.

Hartmanis, J., and H. Lin, eds. "What Is Computer Science and Engineering?" In *Computing the Future: A Broader Agenda for Computer Science and Engineering*, 163–216. Washington, D.C.: National Academy Press, 1992.

Hauser, S. "Computing and Connecting." *Rochester Review* 79, no. 3 (2017): 16–17.

Hayes, B. "Calculating the Weather." *American Scientist* 95, no. 3 (2007).

Hendler, J., N. Shadbolt, W. Hall, T. Berners-Lee, and D. Weitzner. "Web Science: An Interdisciplinary Approach to Understanding the Web." *Communications of the ACM* 51, no. 7 (2008): 60–69.

Hofstatder, D. R. "Review of [Sloman, 1978]." *Bulletin of the American Mathematical Society* 2, no. 2 (1980): 328–39.

Jackendoff, R. *A User's Guide to Thought and Meaning*. Oxford: Oxford University Press, 2012.

Kant, I. *Critique of Pure Reason*. New York: St. Martin's Press, 1781/1787. Norman Kemp Smith translation published 1929.

Khalil, H., and L. S. Levy. "The Academic Image of Computer Science." *ACM SIGCSE Bulletin* 10, no. 2 (1978): 31–33.

Knuth, D. E. "Ancient Babylonian Algorithms." *Communications of the ACM* 15, no. 7 (1972): 671–77.

———. "Computer Programming as an Art." *Communications of the ACM* 17, no. 12 (1974a): 667–73.

———. "Computer Science and Its Relation to Mathematics." *American Mathematical Monthly* 81, no. 4 (1974b): 323–43.

———. "Algorithmic Thinking and Mathematical Thinking." *American Mathematical Monthly* 92, no. 3 (1985): 170–81.

Koen, B. V. "Toward a Definition of the Engineering Method." *European Journal of Engineering Education* 13, no. 3 (1988): 307–15. Reprinted from *Engineering Education* (December 1984): 150–55.

Lamport, L. "How to Write a 21st Century Proof." *Journal of Fixed Point Theory and Applications* 11, no. 1 (2012): 43–63. http://research.microsoft.com/en-us/um/people/lamport/pubs/proof.pdf

Lewis-Kraus, G. "The Great A.I. Awakening." *New York Times Magazine*. December 14, 2016. http://www.nytimes.com/2016/12/14/magazine/the-great-ai-awakening.html

Lindell, S. "Computer Science as a Liberal Art: The Convergence of Technology and Reason." Talk presented at Haverford College. January 24, 2001. http://www.haverford.edu/cmsc/slindell/Presentations/Computer%20Science%20as%20a%20Liberal%20Art.pdf

Loui, M. C. "Computer Science Is an Engineering Discipline." *Engineering Education* 78, no. 3 (1987): 175–78.

———. "Computer Science Is a New Engineering Discipline." *ACM Computing Surveys* 27, no. 1 (1995): 31–32.

———. "Computational Complexity Theory." *ACM Computing Surveys* 28, no. 1 (1996): 47–49.

Mahoney, M. S. *Histories of Computing*, edited by Thomas Haigh. Cambridge, MA: Harvard University Press, 2011.

Maida, A. S., and S. C. Shapiro. "Intensional Concepts in Propositional Semantic Networks." *Cognitive Science* 6 (1982): 291–330.

Marr, D. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. New York: W. H. Freeman, 1982.

McCarthy, J. "A Basis for a Mathematical Theory of Computation." In *Computer Programming and Formal Systems*, edited by P. Braffort and D. Hirschberg. North-Holland, 1963. Page references to PDF version at http://www-formal.stanford.edu/jmc/basis.html

Miller, G. A., E. Galanter, and K. H. Pribram. *Plans and the Structure of Behavior*. Henry Holt, New York, 1960.

Minsky, M. "Computer Science and the Representation of Knowledge." In *The Computer Age: A Twenty Year View*, edited by L. Dertouzos and J. Moses, 392–421. Cambridge, MA: The MIT Press, 1979.

Moor, J. H. "Are There Decisions Computers Should Never Make?" *Nature and System* 1 (1979): 217–29.

Newell, A. "Physical Symbol Systems." *Cognitive Science* 4 (1980): 135–83.

———. "Response: The Models Are Broken, the Models Are Broken." *University of Pittsburgh Law Review* 47 (1985-1986): 1023–31.

Newell, A., A. J. Perlis, and H. A. Simon. "Computer Science." *Science* 157, no. 3795 (1967): 1373–74.

Newell, A., and H. A. Simon. "Computer Science as Empirical Inquiry Symbols and Search." *Communications of the ACM* 19, no. 3 (1976): 113–26.

Oommen, B. J., and L. G. Rueda. "A Formal Analysis of Why Heuristic Functions Work." *Artificial Intelligence* 164, nos. 1-2 (2005): 1–22.

Papert, S. *Mindstorms: Children, Computers, and Powerful Ideas*. New York: Basic Books, 1980.

Perlis, A. "The Computer in the University." In *Management and the Computer of the Future*, edited by M. Greenberger, 181–217. Cambridge, MA: The MIT Press, 1962.

Piccinini, G. *Physical Computation: A Mechanistic Account*. Oxford: Oxford University Press, 2015.

Rapaport, W. J. "How to Make the World Fit Our Language: An Essay in Meinongian Semantics." *Grazer Philosophische Studien*, 14 (1981): 1–21.

———. "Philosophy, Artificial Intelligence, and the Chinese-Room Argument." *Abacus: The Magazine for the Computer Professional*, 3 (1986): 6–17. Correspondence, *Abacus* 4 (Winter 1987): 6–7; 4 (Spring): 5–7; http://www.cse.buffalo.edu/~rapaport/Papers/abacus.pdf

———. "Syntactic Semantics: Foundations of Computational Natural-Language Understanding." In *Aspects of Artificial Intelligence*, edited by J. H. Fetzer, 81–131. Dordrecht, The Netherlands: Kluwer Academic Publishers, 1988.

———. "Understanding Understanding: Syntactic Semantics and Computational Cognition." In *Philosophical Perspectives, Vol. 9: AI, Connectionism, and Philosophical Psychology*, edited by J. E. Tomberlin, 49–88. Atascadero, CA: Ridgeview Publishing, 1995.

———. "How Minds Can Be Computational Systems." *Journal of Experimental and Theoretical Artificial Intelligence* 10 (1998): 403–19.

———. "Implementation Is Semantic Interpretation." *The Monist* 82 (1999): 109–30.

———. "How to Pass a Turing Test: Syntactic Semantics, Natural-Language Understanding, and First-Person Cognition." *Journal of Logic, Language, and Information* 9, no. 4 (2000): 467–90.

———. "What Did You Mean By That? Misunderstanding, Negotiation, and Syntactic Semantics." *Minds and Machines* 13, no. 3 (2003): 397–427.

———. "Implemention Is Semantic Interpretation: Further Thoughts." *Journal of Experimental and Theoretical Artificial Intelligence* 17, no. 4 (2005a): 385–417.

———. "Philosophy of Computer Science: An Introductory Course." *Teaching Philosophy* 28, no. 4 (2005b): 319–41.

———. "Semiotic Systems, Computers, and the Mind: How Cognition Could Be Computing." *International Journal of Signs and Semiotic Systems* 2, no. 1 (2012): 32–71. http://www.cse.buffalo.edu/~rapaport/Papers/Semiotic_Systems,_Computers,_and_the_Mind.pdf

———. "On the Relation of Computing to the World." Forthcoming in *Philosophy and Computing: Essays in Epistemology, Philosophy of Mind, Logic, and Ethics*, edited by T. M. Powers. Springer. Paper based on 2015 IACAP Covey Award talk; preprint available at http://www.cse.buffalo.edu/~rapaport/Papers/covey.pdf

———. *Philosophy of Computer Science*. 2017. Available at http://www.cse.buffalo.edu/~rapaport/Papers/phics.pdf

Rapaport, W. J., and M. W. Kibby. "Contextual Vocabulary Acquisition as Computational Philosophy and as Philosophical Computation." *Journal of Experimental and Theoretical Artificial Intelligence* 19, no. 1 (2007): 1–17.

———. "Contextual Vocabulary Acquisition: From Algorithm to Curriculum." In *Castañeda and His Guises: Essays on the Work of Hector-Neri Castañeda*, edited by A. Palma, 107–50. Berlin: Walter de Gruyter, 2014.

Rapaport, W. J., S. C. Shapiro, and J. M. Wiebe. "Quasi-Indexicals and Knowledge Reports." *Cognitive Science* 21 (1997): 63–107.

Rescorla, M. "The Computational Theory of Mind." In *The Stanford Encyclopedia of Philosophy*, winter 2015 edition, edited by E. N. Zalta. http://plato.stanford.edu/archives/win2015/entries/computational-mind/

Robinson, J. "Logic, Computers, Turing, and von Neumann." In *Machine Intelligence 13: Machine Intelligence and Inductive Learning*, edited by K. Furukawa, D. Michie, and S. Muggleton, 1–35. Oxford: Clarendon Press, 1994.

Romanycia, M. H., and F. J. Pelletier. "What Is a Heuristic?" *Computational Intelligence* 1, no. 2 (1985): 47–58.

Ruff, C. "Computer Science, Meet Humanities: In New Majors, Opposites Attract." *Chronicle of Higher Education* 62, no. 21 (2016): A19.

Samuelson, P., R. Davis, M. D. Kapor, and J. Reichman. "A Manifesto Concerning the Legal Protection of Computer Programs." *Columbia Law Review* 94, no. 8 (1994): 2308–431.

Sayre, K. M. "Intentionality and Information Processing: An Alternative Model for Cognitive Science." *Behavioral and Brain Sciences* 9, no. 1 (1986): 121–65.

Schagrin, M. L., W. J. Rapaport, and R. R. Dipert. *Logic: A Computer Approach*. New York: McGraw-Hill, 1985.

Scott, J., and A. Bundy. "Creating a New Generation of Computational Thinkers." *Communications of the ACM* 58, no. 12 (2015): 37–40.

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

Shannon, C. E. "A Mathematical Theory of Communication." *The Bell System Technical Journal* 27 (1948): 379–423, 623–56.

Shapiro, S. C. "The Cassie Projects: An Approach to Natural Language Competence." In *EPIA 89: 4th Portugese Conference on Artificial Intelligence Proceedings*, edited by J. Martins and E. Morgado, 362–80. Berlin: Springer-Verlag, 1989. Lecture Notes in Artificial Intelligence 390. http://www.cse.buffalo.edu/sneps/epia89.pdf

———. "Computer Science: The Study of Procedures." *Technical report*, Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY, 2001. http://www.cse.buffalo.edu/~shapiro/Papers/whatiscs.pdf

Shapiro, S. C., and W. J. Rapaport. "SNePS Considered as a Fully Intensional Propositional Semantic Network." In *The Knowledge Frontier: Essays in the Representation of Knowledge*, edited by N. Cercone and G. McCalla, 262–315. New York: Springer-Verlag, 1987.

———. "Models and Minds: Knowledge Representation for Natural-Language Competence." In *Philosophy and AI: Essays at the Interface*, edited by R. Cummins and J. Pollock, 215–59. Cambridge, MA: The MIT Press, 1991.

Sheraton, M. "The Elusive Art of Writing Precise Recipes." *The New York Times*. May 2, 1981. http://www.nytimes.com/1981/05/02/style/de-gustibus-the-elusive-art-of-writing-precise-recipes.html

Simon, H. A. "What Computers Mean for Man and Society." *Science* 195, 4283 (1977): 1186–91.

———. "Computational Theories of Cognition." In *The Philosophy of Psychology*, edited by W. O'Donohue and R. F. Kitchener, 160–72. London: SAGE Publications, 1996a.

———. *The Sciences of the Artificial, Third Edition*. Cambridge, MA: The MIT Press, 1996b.

Sloman, A. *The Computer Revolution in Philosophy: Philosophy, Science and Models of Mind*. Atlantic Highlands, NJ: Humanities Press, 1978.

Smith, B.C. "Limits of Correctness in Computers." *ACM SIGCAS Computers and Society* 14–15, nos. 1–4 (1985): 18–26.

Soare, R. I. "Turing Oracle Machines, Online Computing, and Three Displacements in Computability Theory." *Annals of Pure and Applied Logic* 160 (2009): 368–99.

Srihari, S. N. "Beyond C.S.I.: The Rise of Computational Forensics." *IEEE Spectrum*. 2010. http://spectrum.ieee.org/computing/software/beyond-csi-the-rise-of-computational-forensics

Stevens, Jr., Phillip. "Magic." In *Encyclopedia of Cultural Anthropology*, edited by D. Levinson and M. Ember, 721–26. New York: Henry Holt, 1996.

Tedre, M. *The Science of Computing: Shaping a Discipline*. Boca Raton, FL: CRC Press/Taylor and Francis, 2015.

Toussaint, G. "A New Look at Euclid's Second Proposition." *The Mathematical Intelligencer* 15, no. 3 (1993): 12–23.

Turing, A. M. "On Computable Numbers, with an Application to the *Entscheidungsproblem*." *Proceedings of the London Mathematical Society, Ser. 2*, 42 (1936): 230–65.

———. "Computing Machinery and Intelligence." *Mind*, 59, no. 236 (1950): 433–60.

Wiedermann, J. "Simulating the Mind: A Gauntlet Thrown to Computer Science." *ACM Computing Surveys* 31, 3es (1999): Paper No. 16.

Wiesner, J. "Communication Sciences in a University Environment." *IBM Journal of Research and Development* 2, no. 4 (1958): 268–75.

Wing, J. M. "Computational Thinking." *Communications of the ACM* 49, no. 3 (2006): 33–35.

———. "Five Deep Questions in Computing." *Communications of the ACM* 51, no. 1 (2008): 58–60.

———. "Computational Thinking: What and Why?" *The Link (Carnegie-Mellon University)*. November 17, 2010. https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf

Wulf, W. "Are We Scientists or Engineers?" *ACM Computing Surveys* 27, no. 1 (1995): 55–57.

Zemanek, H. "Was ist informatik? (What Is Informatics?)." *Elektronische Rechenanlagen (Electronic Computing Systems)* 13, no. 4 (1971): 157–71.

ARTICLES

Why Think That the Brain Is Not a Computer?

Marcin Miłkowski

INSTITUTE OF PHILOSOPHY AND SOCIOLOGY, POLISH ACADEMY OF SCIENCES

ABSTRACT

In this paper, I review the objections against the claim that brains are computers, or, to be precise, information-processing mechanisms. By showing that practically all the popular objections are either based on uncharitable interpretation of the claim, or simply wrong, I argue that the claim is likely to be true, relevant to contemporary cognitive (neuro)science, and non-trivial.

Computationalism is here to stay. To see why, I will review the reasons why one could think that the brain is not a computer. Although more reasons can be brought to bear

on the issue, my contention is that it's less than likely that they would make any difference. The claim that the brain is a specific kind of an information-processing mechanism, and that information-processing is necessary (even if not sufficient) for cognition, is non-trivial and generally accepted in cognitive (neuro)science. I will not develop the positive view here, however, as it was already stated sufficiently clearly to my tastes in book-length accounts.[1] Instead, I will go through the objections, and show that they all fail just because they make computationalism a straw man.

SOFTWARE AND NUMBER CRUNCHING

One fairly popular objection against computationalism is that there is no simple way to understand the notions of *software* and *hardware* as applied to biological brains. But the software/hardware distinction, popular as the slogan "the mind to the brain is like the software to hardware,"[2] need not be applicable to brains at all for computationalism to be true. There are computers that are not program-controllable: they do not load programs from external memory to internal memory to execute them. The most mundane example of such a computer is a logical gate whose operation corresponds to a logical connective, e.g., disjunction or conjunction. In other words, while it may be interesting to inquire whether there is software in the brain, there may as well be none, and computationalism could still be true. Hence, the objection fails, even if it is repeatedly cited in popular press.

Another intuitive objection, already stated (and defeated) in the 1950s, is that brains are not engaged in number-crunching, while computers, well, compute over numbers. But if this is all computers do, then they don't control missiles, send documents to printers, or display pictures on computer monitors. After all, printing is not *just* number crunching. The objection rests therefore on a mistaken assumption that computers can only compute numerical functions. Computer functions can be defined not only on integer numbers but also on arbitrary symbols,[3] and as physical mechanisms, computers can also control other physical processes.

SYMBOLS AND MEANING

The notion of a symbol is sometimes interpreted to say that symbols in computers are, in some sense, abstract and formal, which would make computers strangely dis-embodied.[4] In other words, the opponents of computationalism claim that it implies some kind of dualism.[5] However, computers are physical mechanisms, and they can be broken, put on fire, and thrown out of the window. These things may be difficult to accomplish with a collection of abstract entities; the last time I tried, I was caught red-handed while committing a simple category mistake. Surely enough, computers are not *just* symbol-manipulators. They do things, and some of the things computers do are not computational. In this sense, computers are physically embodied, not unlike mammal brains. It is, however, a completely different matter whether the symbols in computers mean anything.

One of the most powerful objections formulated against the possibility of Artificial Intelligence is associated with John Searle's Chinese Room thought experiment.[6] Searle claimed to show that running of a computer program is not sufficient for semantic properties to arise, and this was in clear contradiction to what was advanced by proponents of Artificial Intelligence who assumed that it was sufficient to simulate the syntactic structure of representations for the semantic properties to appear; as John Haugeland quipped: "if you take care of syntax, the semantics will take care of itself."[7] But Searle replied: one can easily imagine a person with a special set of instructions in English who could manipulate Chinese symbols and answer questions in Chinese without understanding it at all. Hence, understanding is not reducible to syntactic manipulation. While the discussion around this thought experiment is hardly conclusive,[8] the problem was soon reformulated by Stevan Harnad as "symbol grounding problem":[9] How can symbols in computational machines mean anything?

If symbol grounding problem makes any sense, then one cannot simply assume that symbols in computers mean something just by being parts of computers, or at least they cannot mean anything *outside* the computer so easily (even if they contain instructional information[10]). This is an assumption made also by proponents of causal-mechanistic analyses of physical computation: representational properties are not assumed to necessarily exist in physical computational mechanisms.[11] So, even if Searle is right and there is no semantics in computers, the brain might still be a computer, as computers need no semantics to be computers. Maybe something additional to computation is required for semantics.

Let us make the record straight here. There *is* an important connection between the computational theory of mind and the representational account of cognition: they are more attractive when both are embraced. Cognitive science frequently explains cognitive phenomena by referring to semantic properties of mechanisms capable of information-processing.[12] Brains are assumed to model reality, and these models can be computed over. While this seems plausible to many, it's important to remember than one can remain computationalist without assuming representationalism, or the claim that cognition requires cognitive representation. At the same time, a plausible account of cognitive representation cannot be couched merely in computational terms as long as one assumes that the symbol grounding problem makes sense at least for some computers. To make the account plausible, most theorists appeal to notions of teleological function and semantic information,[13] which are not technical terms of computability theory nor can be reduced to such. So, computers need something special to operate on inherently meaningful symbols.

What made computationalism so strongly connected to cognitive representations was the fact that it offered a solution to the problem of what makes meaning causally relevant. Many theorists claim that just because the syntax in computer programs is causally relevant (or efficacious), so is the meaning. While the wholesale reduction of meaning to syntax is implausible, the computational theory of mind makes it clear that the answer to the question includes the causal role of the syntax of computational vehicles. Still, it is not an objection to computationalism itself that it does

not offer a naturalistic account of meaning. That would be indeed too much.

The debate over the meaning in computers and animals abounds in red herrings, however. One recent example is Robert Epstein's essay.[14] While the essay is ridden with confusion, the most striking mistake is the assumption that computers always represent everything with arbitrary accuracy. Epstein cites the example of how people remember a dollar bill, and assumes that computers would represent it in a photographic manner with all available detail. This is an obvious mistake: representation is useful mostly when it does not convey information about all properties of the represented target (remember that the map of the empire is useful only when it is not exact?[15]). If Epstein is correct, then there are no JPEG files in computers, as they are not accurate, and they are based on lossy compression. And there are no MP3 files. And so on. No assumption of the computational theory of mind says that memory should be understood in terms of the von Neumann architecture, and only some controversial theories suggest that it should.[16]

Epstein also presses the point that people are organisms. Yes, I would also add that water is (mostly) H_2O. It's true but just as irrelevant as Epstein's claim: physical computers are, well, physical, and they may be built in various ways. It's essential that they are physical.

A related objection may be phrased in terms of James J. Gibson's ecological psychology. Ecological psychologists stress that people do not process information, they just pick it up from the environment.[17] This is an interesting idea. But one should make it more explicit what is meant by *information processing* in the computational theory of mind. What kind of information is processed? It should be clear enough that the information need not be semantic, as not all symbols in computers are *about* something. The minimal notion that should suffice for our purposes is the notion of structural information: a vehicle can bear structural information just in case it has at least one degree of freedom, that is, may vary its state.[18] The number of degrees of freedom, or yes-no questions required to exactly describe its current state, is the amount of structural information. As long as there are vehicles with multiple degrees of freedom and they are part of causal processes that cause some other vehicles just like some model of computation describes these processes,[19] there is information processing. This is a very broad notion, as all physical causation implies information transfer and processing in this sense.[20]

Right now it's important to note that the Gibsonian notion of information pickup, interesting as it is, requires vehicles of structural information as well. There needs to be some information out there to be picked up, and organisms have to be so structured to be able to change their state in response to information. Gibsonians could, however, claim that the information is not processed. Frankly, I do not know what is meant by this: for example, Chemero seems to imply that processing amounts to adding more and more layers of additional information, like in Marr's account of vision.[21] Why information processing should require multiple stages of adding more information is beyond me. Even uses of Gibsonian information in, say, simple robots, are clearly computational, and insisting otherwise seems to imply that the dispute is purely verbal. To sum up: the Gibsonian account does not invalidate computationalism at all.

CONSCIOUSNESS

Some people find (some kinds of) consciousness to be utterly incompatible with computationalism, or at least, unexplainable in purely computational terms.[22] The argument is probably due to Leibniz with his thought experiment in *Monadology*.[23] Imagine a brain as huge as a mill, and enter it. Nowhere in the interplay of gears could you find perceptions, or qualitative consciousness. Hence, you cannot explain perception mechanically. Of course, this Leibnizian argument appeals only to some physical features of mechanisms, but some still seem to think that causation has nothing to do with qualitative consciousness. Notice also that the argument, if cogent, is applicable more broadly, not just to computationalism; it is supposed to defeat reductive physicalism or materialism.

For example, David Chalmers claims that while awareness, or the contentful cognitive states and processes, can be explained reductively by appealing to physical processes, there is some qualitative, phenomenal consciousness that escapes all such attempts. But his own positive account (or one of his accounts) is panpsychism, and it states that whenever there is physical information, there is consciousness. Qualitative consciousness. So how is this incompatible with computationalism, again? According to Chalmers, qualitative consciousness supervenes on information with physical necessity (not conceptual one). So be it, but it does not invalidate computationalism, of course.

Notice also that virtually all current theories of consciousness are computational, even the ones that appeal to quantum processes.[24] For example, Bernard Baars offers a computational account in terms of the global workspace theory,[25] David Rosenthal an account in terms of higher-level states,[26] and Giulio Tononi in terms of minimal information integration.[27] Is there any theory of consciousness that is not already computational?

Let us turn to Searle. After all, he suggests that only a non-computational theory of consciousness can succeed. His claim is that consciousness is utterly biological.[28] Fine, but how does this exactly contradict computationalism? You may build a computer of DNA strands,[29] so why claim that it's metaphysically impossible to have a biological computer? Moreover, Searle fails to state which biological powers of brains specifically make them conscious. He just passes the buck to neuroscience. And neuroscience offers computational accounts. Maybe there's a revolution behind the corner, but as things stand, I would not hold my breath for a non-computational account of qualitative consciousness.

TIME AND ANALOG PROCESSING

Proponents of dynamical accounts of cognition stress that Turing machines do not operate in real time. This means that this classical model of computation does not appeal

to real time; instead, it operates with the abstract notion of the computation step. There is no continuous time flow, just discrete clock ticks in a Turing Machine.[30] This is true. But is this an objection against computationalism?

First, there are models of computation that appeal to real time.[31] So one could use such a formalism. Second, the objection seems to confuse the formal model of computation with its physical realization. Physical computers operate in real time, and not all models of computation are made equal; some will be relevant to explaining cognition, and some may be only useful for computability theory. What is required for explanatory purposes is a mechanistically-adequate model of computation that describes all relevant causal processes in the mechanism.[32]

Universal Turing machines are crucial to computability theory. But one could also appeal to models of analog computation if required. These are still understood as computational in computability theory, and some theorists indeed claim that the brain is an analog computer, which is supposed to allow them to compute Turing-incomputable functions.[33] While this is controversial (others claim that brains compute in a more complex fashion[34]), it shows that one cannot dismiss computationalism by saying that the brain is not a digital computer, as Gerald Edelman did.[35] There are analog computers, and an early model of a neural network, Perceptron, was analog.[36] The contention that computers have to be digital is just dogmatic.

ARTIFICIAL INTELLIGENCE

There are a number of arguments with a form:

1. People ψ.
2. Computers will never ψ.

So, artificial intelligence is impossible (or computationalism is false).

This argument is enthymematic, but the conclusion follows with a third assumption: if artificial intelligence is possible, then computers will ψ. The plausibility of the argument varies from case to case, depending on what you fill for ψ. For years, people thought that winning in chess is ψ,[37] but it turned out to be false, which makes the argument instance unsound. So, unless there is a formal proof, it's difficult to treat premise 2 seriously.

So what could be plausibly substituted for ψ? Obviously, not sexual reproduction, even if it is humanly possible. There are many properties of biological organisms that simply seem irrelevant to this argument, including exactly the same energy consumption, having proper names, spatiotemporal location, and so on. The plausible candidate for substitution is some capacity for information-processing. If there is such capacity that humans have but computers cannot, then the argument is indeed cogent.

So what could be the candidate capacity? The classical argument pointed to the human ability to recognize the truth of logical statements that cannot be proven by a computer.[38] It is based on the alleged ability of human beings to understand that some statements are true, which is purportedly impossible only for machines (this argument is based on the Gödel proof of incompleteness of the first-order predicate calculus with basic arithmetic). The problem is that this human understanding has to be non-contradictory and certain. But Gödel has shown that it's undecidable in general whether a given system is contradictory or not; so either the argument states that it's mathematically certain that human understanding of mathematics is non-contradictory, which makes the argument inconsistent (it cannot be mathematically certain because it's undecidable); or it just dogmatically assumes consistency, which means that the argument is implausible, and even unsound because we know that people commit contradictions unknowingly.[39]

Another argument points to common sense. Common sense is a particularly difficult capacity, and the trouble with implementing common sense on machines is sometimes called (somewhat misleadingly) *the frame problem*.[40] Inferential capacities of standard AI programs do not seem to follow the practices known to humans, and that was supposed to hinder progress in such fields as high-quality machine translation,[41] speech recognition (held to be immoral to fund by Weizenbaum[42]), and so on. Even if IBM Watson wins in *Jeopardy!*, one may still think it's not enough. Admittedly, common sense is a plausible candidate in this argument. Notice that even if the proponent of the computational theory of cognition could reject the necessity of building genuine AI that is not based on a computer simulation of human cognitive processes, he or she still has the burden of showing that human common sense can be simulated on a computer. Whether it can or not is still a matter of debate.

COMPUTERS ARE EVERYWHERE (OR DON'T REALLY EXIST)

Still another argument against computationalism brings pretty heavy artillery. The argument has two versions. The first version is the following: at least some plausible theories of physical implementation of computation lead to the conclusion that all physical entities are computational. This stance is called *pancomputationalism*. If this is the case, then the computational theory of mind is indeed trivial, as not only brains are computational, but also cows, black holes, cheese sandwiches, and what not, are computers. However, a pancomputationalist may reply by saying that there are various kinds (and levels) of computation, and brains do not execute all kinds of computation at the same time.[43] So it's not just computation that is specific to brains, but there is some non-trivial kind of computation specific to brains. Only the kind of pancomputationalism that assumes that everything computes all kinds of functions at the same time is catastrophic, as it makes physical computation indeed trivial. But this is what Hilary Putnam claims—he even offered a proof that one can ascribe arbitrary kinds of computation to any open physical system.[44]

Another move is to say that computers do not really exist; they are just in the eyes of beholder. John Searle has made both moves: the beholder decides whether a given physical system is computational, and therefore may make this decision for virtually everything. But the body of work

on physical computation in the last decade or so has been focused on showing why Putnam and Searle were wrong.[45] The contemporary consensus is that computational models can adequately describe causal connections in physical systems, and that these models can be also ascribed wrongly. In other words, computational models are not different in kind from any mathematical model used in science. If they are mere subjective metaphors and don't describe reality, then mathematical models in physics are subjective as well.[46]

Intuitively, arguments presented by Searle and Putnam are wrong for a very simple reason: nobody would buy a new computer if it was just easier to think that an old computer simply implemented new software. I could stare at my old laptop and think that it's a brand new smartphone. It's obvious that it doesn't work this way. Therefore, there must be a flaw in these arguments somewhere, and even if the technicalities involved are indeed interesting, they fail to establish the conclusion.

A popular strategy to defeat triviality arguments is to show that it is *ad hoc*: the ascriptions of computational states to physical systems wouldn't support relevant counterfactuals.[47] In other words, they couldn't, for example, accurately predict what kind of computation would run on a physical system, were things slightly different. While this is intuitive, I have argued that one can strengthen the triviality strategies to deal with counterfactuals.[48] As long as one is poised to predict the evolution of a physical process, one can invent a computational ascription. Thus, instead, one should look for a systematic solution that presupposes that computational models are not different in kind from other causal models in science. This is the move recommended by David Chalmers, who has stressed that computational models should be understood causally.[49] However, his strategy requires all computational models to be rephrased to use his favorite mathematical model of computation, combinatorially structured finite-state machine (CFSA), and then matched to a causal structure of a physical system. But rephrasing has an important disadvantage: the states of an original model of computation may turn out to be causally inefficacious. This is why, in reply to Chalmers, I suggested that computation should be modeled *directly* in a mechanistically-adequate model of computation whose causal organization matches the organization of a physical mechanism, and appeal to standard explanatory norms.[50] The norms of mechanistic explanation, which should be followed when explaining a computational system causally, are sufficient to block triviality arguments. (For example, ascriptions will turn out to be extremely non-parsimonious, and will not offer any new predictions except the ones already known from a physical description of a system, which suggests that the model is based on so-called over-fitting.)

All in all, triviality arguments required theorists to spell out the account of physical computation much more clearly but are not a real danger to computationalism. This is not to say that more often than not, empirical evidence is insufficient to decide between vastly different hypotheses about the computational organization of a given mechanism. But again, this is not in any way special for computational hypotheses, since theories are generally underdetermined by evidence.

CONCLUSION

Let me wrap up. In this paper, I have listed and summarized a number of arguments against computationalism. The only objection that does not seem to be implausible at the first glance is the one that states that common sense is impossible or extremely difficult to implement on a machine. However, more and more commonsensical capacities are being implemented on machines. For example, in the 1990s and early 2000s, I used to work as a technical translator for software companies. We used to laugh at machine translation, and nobody would use it professionally. But it's the machine that translates the Microsoft Knowledge Base, which was extremely difficult for professionals to deal with. While the quality of machine translation is still behind the best human beings for complex literary translations, it is no longer something that translators laugh at. We use machine translation at work and merely post-edit it.

The point is that there's no good reason to think that the brain is not a computer. But it's not just a computer. It is, of course, physically embedded in its environment and interacts physically with it with its body, and for that, it also needs a peripheral nervous system[51] and cognitive representations. But there's nothing that denies computationalism here. Most criticisms of computationalism therefore fail, and sticking to them is probably a matter of ideology rather than rational debate.

ACKNOWLEDGEMENTS

The work on this paper was funded by a National Science Centre (Poland) research grant under the decision DEC-2014/14/E/HS1/00803. The author wishes to thank Piotr Bołtuć, Tomasz Korbak, Martin Hughes, Panu Raatikainen, and Błażej Skrzypulec, as well as the anonymous referees of this paper for their comments.

NOTES

1. Piccinini, *Physical Computation: A Mechanistic Account*; Miłkowski, *Explaining the Computational Mind*.
2. Block, "The Mind as the Software of the Brain"; Piccinini, "The Mind as Neural Software?"
3. Newell, "Physical Symbol Systems."
4. Lakoff,*Women, Fire, and Dangerous Things*; Barrett, "Why Brains Are Not Computers, Why Behaviorism Is Not Satanism, and Why Dolphins Are Not Aquatic Apes"; Barrett, Pollet, and Stulp, "From Computers to Cultivation: Reconceptualizing Evolutionary Psychology."
5. Searle, "Is the Brain's Mind a Computer Program?"
6. Searle, "Minds, Brains, and Programs."
7. Haugeland,*Artificial Intelligence: The Very Idea*, 106.
8. Preston and Bishop, *Views into the Chinese Room: New Essays on Searle and Artificial Intelligence*.
9. Harnad, "The Symbol Grounding Problem."
10. Fresco and Wolf, "The Instructional Information Processing Account of Digital Computation."
11. Fresco, "Explaining Computation Without Semantics: Keeping It Simple"; Piccinini, "Computation without Representation"; Miłkowski, *Explaining the Computational Mind*.

12. Shagrir, "Brains as Analog-Model Computers."
13. Millikan, *Language, Thought, and Other Biological Categories: New Foundations for Realism*; Dretske, "Misrepresentation"; Bickhard, "The Interactivist Model"; Cummins and Roth, "Meaning and Content in Cognitive Science."
14. Epstein, "The Empty Brain."
15. Borges, *A Universal History of Infamy*.
16. Gallistel and King, *Memory and the Computational Brain*.
17. Gibson, *The Ecological Approach to Visual Perception*; cf. Chemero, "Information for Perception and Information Processing."
18. MacKay, *Information, Mechanism, and Meaning*.
19. Miłkowski, "Computational Mechanisms and Models of Computation."
20. Collier, "Causation Is the Transfer of Information."
21. Chemero, "Information for Perception and Information Processing," 584; cf. Marr, *Vision. A Computational Investigation into the Human Representation and Processing of Visual Information*.
22. Chalmers, *The Conscious Mind: In Search of a Fundamental Theory*.
23. Leibniz, *The Monadology*.
24. Hameroff, "The Brain Is Both Neurocomputer and Quantum Computer."
25. Baars, *A Cognitive Theory of Consciousness*; cf. also Dennett, *Sweet Dreams. Philosophical Obstacles to a Science of Consciousness*.
26. Rosenthal, *Consciousness and Mind*; cf. Cleeremans, "Computational Correlates of Consciousness."
27. Tononi, "An Information Integration Theory of Consciousness."
28. Searle, *The Rediscovery of the Mind*.
29. Zauner and Conrad, "Parallel Computing with DNA: Toward the Anti-Universal Machine."
30. Bickhard and Terveen, *Foundational Issues in Artificial Intelligence and Cognitive Science: Impasse and Solution*; Wheeler, *Reconstructing the Cognitive World*.
31. Nagy and Akl, "Computations with Uncertain Time Constraints: Effects on Parallelism and Universality."
32. Miłkowski, "Computational Mechanisms and Models of Computation."
33. Siegelmann, "Analog Computation via Neural Networks."
34. Piccinini and Bahar, "Neural Computation and the Computational Theory of Cognition."
35. Edelman, *Bright Air, Brilliant Fire*.
36. Rosenblatt, "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain."
37. Dreyfus, *What Computers Still Can't Do: A Critique of Artificial Reason*.
38. Lucas, "Minds, Machines and Gödel"; Penrose, *The Emperor's New Mind*.
39. Krajewski, "On Gödel's Theorem and Mechanism: Inconsistency or Unsoundness Is Unavoidable in Any Attempt to 'Out-Gödel' the Mechanist"; Putnam, "Minds and Machines."
40. Dreyfus, *What Computers Can't Do: A Critique of Artificial Reason*; Wheeler, *Reconstructing the Cognitive World*.
41. Bar-Hillel, "A Demonstration of the Nonfeasibility of Fully Automatic High Quality Translation."
42. Weizenbaum, *Computer Power and Human Reason: From Judgment to Calculation*.
43. Miłkowski, "Is Computationalism Trivial?"
44. Putnam, *Representation and Reality*; Searle, *The Rediscovery of the Mind*.
45. Chalmers, "A Computational Foundation for the Study of Cognition"; Piccinini, *Physical Computation: A Mechanistic Account*; Miłkowski, *Explaining the Computational Mind*; Shagrir, "Towards a Modeling View of Computing"; Scheutz, "When Physical Systems Realize Functions…"; Chrisley, "Why Everything Doesn't Realize Every Computation"; Copeland, "What Is Computation?"
46. McDermott, *Mind and Mechanism*.
47. Copeland, "What Is Computation?"
48. Miłkowski, *Explaining the Computational Mind*.
49. Chalmers, "A Computational Foundation for the Study of Cognition."
50. Miłkowski, "Beyond Formal Structure: A Mechanistic Perspective on Computation and Implementation."
51. Aranyosi, *The Peripheral Mind*.

REFERENCES

Aranyosi, István. *The Peripheral Mind: Philosophy of Mind and the Peripheral Nervous System*. New York, NY: Oxford University Press, 2013.

Baars, Bernard J. *A Cognitive Theory of Consciousness*. Cambridge/New York: Cambridge University Press, 1988.

Bar-Hillel, Yehoshua. "A Demonstration of the Nonfeasibility of Fully Automatic High Quality Translation." In *Language and Information*, 174–79. Reading, MA: Addison-Wesley, 1964.

Barrett, Louise. "Why Brains Are Not Computers, Why Behaviorism Is Not Satanism, and Why Dolphins Are Not Aquatic Apes." *The Behavior Analyst*, 2015, 1–15. doi:10.1007/s40614-015-0047-0.

Barrett, Louise, Thomas V. Pollet, and Gert Stulp. "From Computers to Cultivation: Reconceptualizing Evolutionary Psychology." *Frontiers in Psychology* 5 (January 2014). doi:10.3389/fpsyg.2014.00867.

Bickhard, Mark H. "The Interactivist Model." *Synthese* 166, no. 3 (2008): 547–91. doi:10.1007/s11229-008-9375-x.

Bickhard, Mark H., and L. Terveen. *Foundational Issues in Artificial Intelligence and Cognitive Science: Impasse and Solution*. North-Holland, 1995.

Block, Ned. "The Mind as the Software of the Brain." In *An Invitation to Cognitive Science*, edited by D. Osherson, L. Gleitman, and S. Kosslyn. Cambridge, MA: The MIT Press, 1995.

Borges, Jorge Luis. *A Universal History of Infamy*. Translated by Norman Thomas di Giovanni. Harmondsworth: Penguin, 1981.

Chalmers, David J. "A Computational Foundation for the Study of Cognition." *Journal of Cognitive Science*, no. 12 (2011): 325–59.

———. *The Conscious Mind: In Search of a Fundamental Theory*. New York: Oxford University Press, 1996.

Chemero, Anthony. "Information for Perception and Information Processing." *Minds and Machines* 13 (2003): 577–88.

Chrisley, Ronald L. "Why Everything Doesn't Realize Every Computation." *Minds and Machines* 4, no. 4 (November 1994): 403–20. doi:10.1007/BF00974167.

Cleeremans, Axel. "Computational Correlates of Consciousness." *Progress in Brain Research* 150 (2005): 81–98. doi:10.1016/S0079-6123(05)50007-4.

Collier, John D. "Causation Is the Transfer of Information." In *Causation, Natural Laws and Explanation*, edited by Howard Sankey, 279–331. Dordrecht: Kluwer, 1999.

Copeland, B. Jack. "What Is Computation?" *Synthese* 108, no. 3 (1996): 335–59.

Cummins, Robert, and Martin Roth. "Meaning and Content in Cognitive Science." In *Prospects for Meaning*, edited by Richard Schantz, 365–82. Berlin and New York: de Gruyter, 2012.

Dennett, Daniel C. *Sweet Dreams. Philosophical Obstacles to a Science of Consciousness*. Cambridge, MA: The MIT Press, 2005.

Dretske, Fred I. "Misrepresentation." In *Belief: Form, Content, and Function*, edited by Radu Bogdan, 17–37. Oxford: Clarendon Press, 1986.

Dreyfus, Hubert. *What Computers Can't Do: A Critique of Artificial Reason*. New York: Harper and Row, Publishers, 1972.

———. *What Computers Still Can't Do: A Critique of Artificial Reason*. Cambridge MA: The MIT Press, 1979.

Edelman, Gerald M. *Bright Air, Brilliant Fire: On the Matter of the Mind*. New York, NY: BasicBooks, 1992.

Epstein, Robert. "The Empty Brain." *Aeon*, May 18, 2016. https://aeon.co/essays/your-brain-does-not-process-information-and-it-is-not-a-computer

Fresco, Nir. "Explaining Computation Without Semantics: Keeping It Simple." *Minds and Machines* 20, no. 2 (June 2010): 165–81. doi:10.1007/s11023-010-9199-6.

Fresco, Nir, and Marty J. Wolf. "The Instructional Information Processing Account of Digital Computation." *Synthese* 191, no. 7 (September 2013): 1469–92. doi:10.1007/s11229-013-0338-5.

Gallistel, C. R., and Adam Philip King. *Memory and the Computational Brain*. Chichester: Wiley-Blackwell, 2010.

Gibson, James J. *The Ecological Approach to Visual Perception*. Hove: Psychology Press, 1986.

Hameroff, Stuart R. "The Brain Is Both Neurocomputer and Quantum Computer." *Cognitive Science* 31 (2007): 1035–45.

Harnad, Stevan. "The Symbol Grounding Problem." *Physica D* 42 (1990): 335–46.

Haugeland, John. *Artificial Intelligence: The Very Idea*. Cambridge, MA: The MIT Press, 1985.

Krajewski, Stanisław. "On Gödel's Theorem and Mechanism: Inconsistency or Unsoundness Is Unavoidable in Any Attempt to 'Out-Gödel' the Mechanist." *Fundamenta Informaticae* 81, no. 1 (January 2007): 173–81.

Lakoff, George. *Women, Fire, and Dangerous Things: What Categories Reveal about the Mind*. Chicago: University of Chicago Press, 1987.

Leibniz, Gottfried Wilhelm. *The Monadology*. Translated by Robert Latta. Raleigh, NC; Boulder, CO: Alex Catalogue; NetLibrary, 1991.

Lucas, J. R. "Minds, Machines and Gödel." *Philosophy* 9, no. 3 (April 1961): 219–27.

MacKay, Donald MacCrimmon. *Information, Mechanism, and Meaning*. Cambridge, MA: The MIT Press, 1969.

Marr, David. *Vision. A Computational Investigation into the Human Representation and Processing of Visual Information*. New York: W. H. Freeman and Company, 1982.

McDermott, Drew V. *Mind and Mechanism*. Cambridge, MA: The MIT Press, 2001.

Miłkowski, Marcin. "Beyond Formal Structure: A Mechanistic Perspective on Computation and Implementation." *Journal of Cognitive Science* 12, no. 4 (2011): 359–79.

———. "Computational Mechanisms and Models of Computation." *Philosophia Scientae* 18, no. 18–3 (2014): 215–28. doi:10.4000/philosophiascientiae.1019.

———. *Explaining the Computational Mind*. Cambridge, MA: The MIT Press, 2013.

———. "Is Computationalism Trivial?" In *Computation, Information, Cognition – The Nexus and the Liminal*, edited by Gordana Dodig Crnkovic and Susan Stuart, 236–46. Newcastle: Cambridge Scholars Press, 2007. http://marcinmilkowski.pl/downloads/is-computationalism-trivial.pdf

Millikan, Ruth Garrett. *Language, Thought, and Other Biological Categories: New Foundations for Realism*. Cambridge, MA: The MIT Press, 1984.

Nagy, Naya, and Selim Akl. "Computations with Uncertain Time Constraints: Effects on Parallelism and Universality." In *Unconventional Computation*, edited by Cristian Calude, Jarkko Kari, Ion Petre, and Grzegorz Rozenberg, 6714:152–63. Lecture Notes in Computer Science. Springer Berlin/Heidelberg, 2011. http://dx.doi.org/10.1007/978-3-642-21341-0_19

Newell, Allen. "Physical Symbol Systems." *Cognitive Science: A Multidisciplinary Journal* 4, no. 2 (1980): 135–83. doi:10.1207/s15516709cog0402_2.

Penrose, Roger. *The Emperor's New Mind*. London: Oxford University Press, 1989.

Piccinini, Gualtiero. "Computation without Representation." *Philosophical Studies* 137, no. 2 (September 2008): 205–41. doi:10.1007/s11098-005-5385-4.

———. *Physical Computation: A Mechanistic Account*. Oxford: Oxford University Press, 2015.

———. "The Mind as Neural Software? Understanding Functionalism, Computationalism, and Computational Functionalism." *Philosophy and Phenomenological Research* 81, no. 2 (September 1, 2010): 269–311. doi:10.1111/j.1933-1592.2010.00356.x.

Piccinini, Gualtiero, and Sonya Bahar. "Neural Computation and the Computational Theory of Cognition." *Cognitive Science* 37, no. 3 (April 2013): 453–88. doi:10.1111/cogs.12012.

Preston, John, and Mark Bishop. *Views into the Chinese Room: New Essays on Searle and Artificial Intelligence*. Oxford; New York: Clarendon Press, 2002.

Putnam, Hilary. "Minds and Machines." In *Dimensions of Mind*, edited by Sidney Hook. New York University Press, 1960.

———. *Representation and Reality*. Cambridge, MA: The MIT Press, 1991.

Rosenblatt, F. "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain." *Psychological Review* 65, no. 6 (1958): 386–408. doi:10.1037/h0042519.

Rosenthal, David. *Consciousness and Mind*. Oxford, New York: Oxford University Press, 2005.

Scheutz, Matthias. "When Physical Systems Realize Functions…." *Minds and Machines* 9, no. 2 (1996): 1–34. doi:10.1023/A:1008364332419.

Searle, John R. "Is the Brain's Mind a Computer Program?" *Scientific American*, no. January (1990): 26–31.

———. "Minds, Brains, and Programs." *Behavioral and Brain Sciences* 3, no. 03 (February 1980): 1–19. doi:10.1017/S0140525X00005756.

———. *The Rediscovery of the Mind*. Cambridge, MA: The MIT Press, 1992.

Shagrir, Oron. "Brains as Analog-Model Computers." *Studies In History and Philosophy of Science Part A* 41, no. 3 (September 2010): 271–79. doi:10.1016/j.shpsa.2010.07.007.

———. "Towards a Modeling View of Computing." In *Information and Computation*, edited by Gordana Dodig-Crnkovic and Mark Burgin. Singapore: World Scientific Publishing, 2010.

Siegelmann, H. "Analog Computation via Neural Networks." *Theoretical Computer Science* 131, no. 2 (September 1994): 331–60. doi:10.1016/0304-3975(94)90178-3.

Tononi, Giulio. "An Information Integration Theory of Consciousness." *BMC Neuroscience* 5, no. 1 (November 2004). doi:10.1186/1471-2202-5-42.

Weizenbaum, Joseph. *Computer Power and Human Reason: From Judgment to Calculation*. San Francisco: W. H. Freeman, 1976.

Wheeler, Michael. *Reconstructing the Cognitive World*. Cambridge, MA: The MIT Press, 2005.

Zauner, Klaus-Peter, and Michael Conrad. "Parallel Computing with DNA: Toward the Anti-Universal Machine." In *Parallel Problem Solving from Nature - PPSN IV*, edited by Hans-Michael Voigt, Werner Ebeling, Ingo Rechenberg, and Hans-Paul Schwefel, 1141:696–705. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 1996. http://dx.doi.org/10.1007/3-540-61723-X_1033

From Biological to Synthetic Neurorobotics Approaches to Understanding the Structure Essential to Consciousness (Part 2)

Jun Tani[1]
DEPARTMENT OF ELECTRICAL ENGINEERING, KOREAN ADVANCED INSTITUTE OF SCIENCE AND TECHNOLOGY (KAIST), TANI1216JP@GMAIL.COM

Jeff White
COMPUTATIONAL NEUROSYSTEM LABORATORY, KAIST

ABSTRACT
This paper reviews research in "predictive coding" that ultimately provides a platform for testing competing theses about specific dynamics inherent in consciousness embodied in both biological and artificial systems.

1 INTRODUCTION
We have been left with a big challenge, to articulate consciousness and also to prove it in an artificial agent against a biological standard. After introducing Boltuc's h-consciousness in the last paper, we briefly reviewed some salient neurology in order to sketch less of a standard than a series of targets for artificial consciousness, "most-consciousness" and "myth-consciousness." With these targets on the horizon, we began reviewing the research program pursued by Jun Tani and colleagues in the isolation of the formal dynamics essential to either. In this paper, we describe in detail Tani's research program, in order to make the clearest case for artificial consciousness in these systems. In the next paper, the third in the series, we will return to Boltuc's naturalistic non-reductionism in light of the neurorobotics models introduced (alongside some others), and evaluate them more completely.

1.1 PREDICTIVE CODING
In this section, we will review a research program into artificial consciousness that demonstrates the potential for computational experiments to isolate the formal dynamics of consciousness including the sense of time. Our focus is on the capacity for agents like human beings to project and to act towards possible futures by reflecting on the past. Studies in biological cognition have set out this capacity in terms of "predictive coding."[2] With predictive coding, the results of actions—common "experience"—are integrated into an agent in terms of "prediction error."

Prediction error informs the agent about how far from an intended target a prior action has led it, with the agent's implicit aim being the minimization of this error signal. That said, minimization of error is not absolute. Optimizing for long-term ends may result in a relative detachment from the immediate perceptual reality, and conversely overt attention on immediate rewards may result in mounting error over the long run. Because predictive coding makes this form of future-oriented proactive agency based on effortful past regression possible within a mathematically embodied agent, it offers a promising formal framework within which the relationship between the subjective mind and the objective world may be instantiated in an artificial agent.

Predictive coding is an important development in artificial consciousness research in two important ways. One, it provides a direct way to model subjective intention within the objective world. And two, it provides an equally direct way to project back the reality of the objective world as perceived by and as consequent on the actions of embodied and embedded cognitive agents.[3] The result is a fully accessible dynamical mirror into the operations essential to consciousness in more complex systems, a promise that merely biological approaches to the study of consciousness cannot match. Tani was the first to successfully instantiate predictive coding in artificial agents, e.g., robots, in a deterministic domain, i.e., where intended outcomes are stable attractors.[4] Alternatively, Friston explored Bayesian predictive coding in a probabilistic domain and generalized it under the name of the "free energy minimization principle" (FEMP).[5]

In the next section, we will briefly review a dynamic neural network model, the recurrent neural network (RNN),[6] because it is a basic component of contemporary intelligent systems, and central to Tani's deterministic dynamics which is the subject of the subsequent section. This review should serve as a primer on the dynamic system's approach to embodied cognition. After reviewing Tani and colleagues' formulation using RNN models, we will examine Bayesian predictive coding as formulated by Friston and colleagues.

2. PREDICTIVE CODING IN DYNAMIC NEURAL NETWORK MODELS

2.1 THE RNN MODEL
The essential characteristic of the RNN[7] is that it can generate temporal sequence patterns as targets embedded in its internal dynamic structure. It "learns" to imitate exemplar sequence patterns, and when properly organized even to creatively compose its own[8] by extracting underlying regularity. An example of an RNN is shown in Figure 1. This figure shows an RNN used in the predictive learning scheme to be described later (section 2.2).

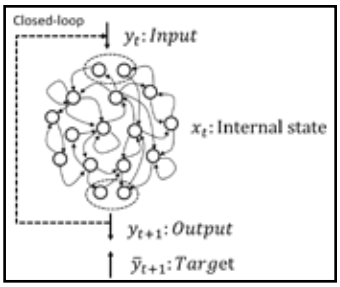

Figure 1. RNN model of predictive learning with teaching target. The dotted line represents a closed-loop copy from the output to the input.

An RNN consists of a set of neural units including input units representing the input state, internal (context) units representing the internal state and output units representing the output state. These are variously interconnected by synaptic connectivity weights. These connections can be unidirectional, bidirectional, or recurrent. The time

development of each neural unit output activation in discrete time can be written as:

$$u^i_{t+1} = \sum^N_j w_{ij} a^j_t + b^i \quad \text{(1-a)}$$

$$a^i_t = f(u^i_t) \quad \text{(1-b)}$$

Where u^i_t is the internal state of the ith neural unit at time step *t*, a^i_t is its output activation, w_{ij} is synaptic connection weight from the *j*th unit to the *i*th unit, b^i is the bias of the *i*th unit. f() is a sigmoid function. Over time, the neural activation of the whole network can generate different types of dynamic attractor patterns depending on the synaptic weights adopted in the network. Figure 2 shows typical attractors including a fixed point attractor, a limit cycle attractor, a limit torus, and chaotic attractor.

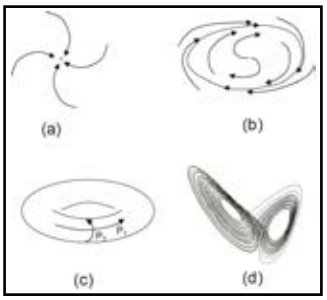

Figure 2. Four different types of attractors. (a) fixed point attractor, (b) limit cycle attractor, (c) limit torus characterized by two periodicities P1 and P2 which form an irrational fraction, and (d) chaotic attractor.

The simplest attractor is a fixed point attractor in which all dynamic states converge to a point (Fig. 2 (a)). The second one is a limit cycle attractor (Fig. 2 (b)) in which the trajectory converges to a cyclic oscillation pattern with constant periodicity. The third one is a limit torus that appears when there is more than one frequency involved in the periodic trajectory of the system and two of these frequencies form an irrational fraction. In this case, the trajectory is no longer closed and it exhibits quasi-periodicity (Fig. 2(c)). The fourth one is a chaotic attractor in which the trajectory exhibits infinite periodicity and thereby forms fractal structures (Fig. 2 (d)).

These different types of attractor dynamics can account for the autonomous generation of different types of agent action patterns. For example, fixed point attractor dynamics account for a hand reaching movement, from an arbitrary hand posture to its end point, while limit cycle attractor dynamics account for a rhythmical hand waiving pattern with a certain periodicity, and chaotic attractor dynamics account for non-periodic, seemingly random movement.

RNNs can learn to generate such attractor dynamics through predictive learning. Each specific attractor pattern can be developed in an RNN by optimizing the synaptic weights and biases through a process of error minimization. In predictive learning, the network receives current time step perceptual input and outputs a prediction of the next time step (see Fig.1). Error is computed between the predicted output and the target (e.g., teaching exemplar), and synaptic weights and biases are updated in the direction of minimizing this error using error back-propagation through time (BPTT).[9] After learning, the RNNs internal dynamics converge on a stable pattern, and the learned attractor can be generated from a given initial state through "closed-loop" (off-line) operation in which the predicted output of the current time step is copied to the input of the next time step in a closed-loop (see the dotted line in Fig.1). This closed-loop operation corresponds to mental simulation, as will be described later sections.

An RNN can be regarded as a dynamical system with adaptive parameters including synaptic weights and biases which can be described in the following generalized form

$$x_{t+1} = F(x_t, w) \quad \text{(2a)}$$

$$y_{t+1} = G(x_{t+1}, W) \quad \text{(2b)}$$

In these expressions, x_t and y_t represent the current internal state and the output state, respectively, and w stands for the adaptive parameter. The internal state x_t is important, because it represents the current context or situation for the system as a whole and develops by means of an iterative learning process. The system can exhibit contextual information processing through which the output of the system reflects not merely the immediately perceived inputs but the context accumulated over past experiences of inputs. Formally speaking, this system embodies temporality, entrained according to patterns that extend beyond the immediate context and, as we shall see, reaches—even creatively, and inferentially—toward goal states.

The conventional RNN model can learn to generate only a single attractor pattern except special cases of developing multiple attractors. So, a natural question arises: How can the model be advanced such that it can learn to generate multiple attractor patterns, each specific to a different context? This question motivated an investigation into the possibility of applying the framework of predictive coding in the advancement of RNN models, as described next.

2.2 MIXRNNS AND RNNPB

Tani and colleagues investigated how a network model can retrieve and generate a particular sequential pattern from long-term memory of multiple patterns. Two versions of RNN models resulted, namely a mixture of RNN experts (MixRNNs)[10] and a recurrent neural network with parametric bias (RNNPB).[11] MixRNNs use a local representation scheme, and RNNPBs use a distributed representation scheme, in order to learn to generate and to recognize sequences of primitive action patterns. Moreover, these movement patterns are temporal patterns requiring active self-entrainment through online information by another's live and more-or-less similarly embodied example, recalling the mechanism of "mirror neurons."[12] In this section, we look more closely at how the MixRNN and the RNNPB capture aspects of consciousness typically associated with more complex biological systems.

MixRNNs[13] consist of sets of local RNNs internally associated through gates where the global output of the whole network model is computed as the weighted sum of the gate opening ratio for all local RNN outputs (see Figure 3).

During learning, local RNNs compete to predict the next perceptual state as the gate opens most for the local RNN with the least prediction error. Because the learning rate of

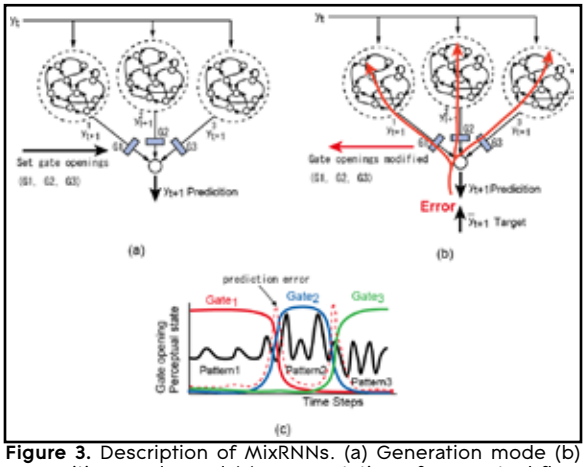

Figure 3. Description of MixRNNs. (a) Generation mode (b) recognition mode, and (c) segmentation of perceptual flow into a sequence of chunked sub-patterns by inferencing gate openings.

each RNN is proportional to the gate opening ratio, the more that the gate of a particular RNN opens, the more this local RNN is able to learn the current perceptual sequence pattern. The goal of learning is to obtain optimal synaptic weights for all modular RNNs as well as optimal openings of the all gates at each time step, and by "optimal" we mean those which minimize the reconstruction error between the global output and the target output.[14] Through a competitive learning process, i.e., error regression training with BPTT for the optimal gate opening sequence between RNNs, as well as for optimal synaptic weights in all local RNNs, each local RNN becomes an expert for a particular perceptual sequence pattern. Intuitively then, MixRNNs can learn a set of frequently apparent primitive patterns with each consolidated in a corresponding local RNN simply through the iterative and collective experience of those patterns.

After learning, a MixRNN model can generate a particular intended perceptual sequence by opening the gate of the corresponding RNN expert (Fig. 3(a)). In this way, current gate openings represent the current top-down intention designating the pattern to be generated. Additionally, MixRNNs can recognize a given perceptual sequence pattern through competition between local RNNs by reconstructing the target pattern with the least error by means of the error regression scheme optimizing the gate openings (Fig. 3 (b), with synaptic weights fixed in this case). When error is minimal, a gate associated with a particular local RNN opens in a winner-take-all manner, and the target pattern is recognized as belonging to this expert RNN. In other words, the target pattern of the current perception can be recognized by means of reconstructing it in a particular local RNN with minimum error whereby the current gate opening states represent the inferred intention.

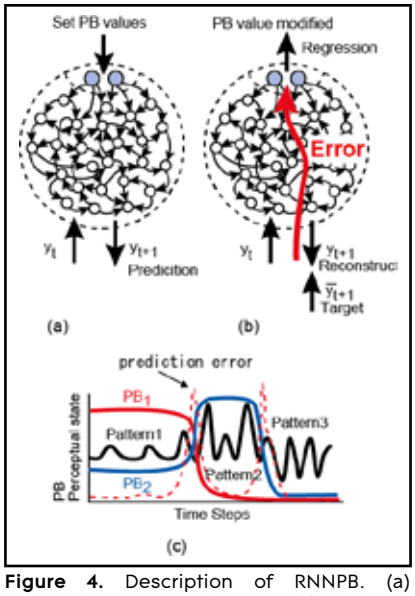

Figure 4. Description of RNNPB. (a) Generation mode, (b) recognition mode, and (c) segmentation of perceptual flow by PB vector into chunked patterns.

When the currently perceived sequential pattern changes, gate opening is shifted toward minimizing prediction error arising at this moment. An important point here is that the continuous perceptual flow is segmented into chunks by means of gate openings during these moments. Tani and Nolfi argue that this suddenly required effort for minimizing the error by inferring appropriate gate openings should accompany momentary consciousness.[15] Next, we look at a further advance on RNNs in this direction, the recurrent neural network with parametric bias, or RNNPB.

The RNNPB[16] is a single RNN model employing parametric bias (PB) units (see Figure 4).

PB represents the current intention as it projects a particular perceptual sequential pattern onto the external world, analogous to the gate dynamics in MixRNNs. PB does this job by playing the role of bifurcation parameter modulating the dynamical structure of the RNN.

In simple terms, an RNNPB learns to predict or generate a set of perceptual sequence patterns associated with corresponding PB vector values. During learning, the optimal synaptic weights for all different sequence patterns as well as the optimal PB vector value for each sequence pattern can be obtained. After learning, an RNNPB can generate a learned perceptual sequence pattern by adopting the PB with the corresponding vector value (Fig. 4(a)). It can also recognize a perceptual sequence pattern given as a target by inferring the optimal PB vector by way of which the target sequence can be reconstructed and output with the minimum error (Fig. 4(b)). Fig. 4(c) shows how the continuous perceptual stream can be segmented into a sequence of prior-learned patterns in terms of attractor dynamics by tracking modulations in PB vector bifurcation parameters at each time step.

In the end, switching between chunks in the RNNPB is analogous to the segmentation mechanism employed in MixRNNs which use gates between local RNNs to recruit the appropriate expert or combination of experts given immediate perceptual reality. One limitation common to both is that each consists of a single level. But, when they are organized into a hierarchy, they can exhibit higher-order cognitive competencies such as creative compositionality.

Such an extension is the subject of the next section.

2.3 FUNCTIONAL HIERARCHY IN THE MTRNN

Both MixRNNs and the RNNPB have been developed into multiple-level architectures.[17] The basic idea is that higher levels attempt to control lower levels by projecting control parameters (such as gate openings or PB vector modulations) onto lower levels based on current higher

order intention. And in turn, during normal operation the prediction error generated against the perceptual reality in the lower level is back-propagated to the higher level, where the control parameters for the lower level as well as the intention state in the higher level is adjusted in the direction of minimizing the error, i.e., by conforming to that state which would have resulted in least error.[18]

Tani and Nolfi demonstrate that hierarchically organized MixRNNs can learn to generate and recognize a set of sequential combinations of movement primitives in a simulated indoor robot navigation space.[19] The analysis showed that a set of chunks related to movement primitives such as turning to the right/left at a corner, going straight along a corridor, and passing through a T-branch developed in local lower level RNNs, while different sequential combinations of these primitives developed in the higher-level RNNs, e.g., traveling through different rooms. When the simulated robot, for example, turns left at a corner from a straight corridor in a particular room, the continuous perceptual flow of its range sensor is segmented into the corresponding two movement primitives in the lower level. On the other hand, when it travels from a familiar room to another, segmentation related to the room transition can take a place in the higher level.

Tani achieved similar results in a real robotic arm with a similarly hierarchically organized RNNPB, which was able to deal with primitives and their sequential combinations during a simple object manipulation task. It is important to note that what begins as raw experience of the continuous perceptual flow becomes a manipulable object for the higher level after segmentation into chunks. Thus, the hierarchical structure adopted by Tani enables the objectification of perceptual experience, as will be described in greater detail later.[20]

Building on this work in hierarchically organized RNNs, Yamashita and Tani[21] demonstrated the learning of compositional action generation by a humanoid robot employing a novel multiple timescale RNN (MTRNN) (Figure 5). This MTRNN model uses multiple timescale constraints, with higher-level activity constrained by slower timescale dynamics, and with lower level activity proceeding according to faster timescale dynamics. The basic idea is that higher-level information processing becomes more abstract as constrained by its slower dynamics, whereas lower level information processing is more sensitive to immediate details as constrained by faster dynamics.

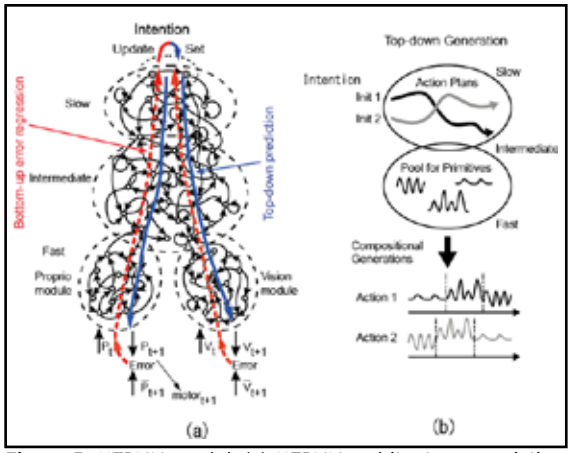

Figure 5. MTRNN model. (a) MTRNN architecture consisting of 3 levels, and (b) its top-down compositional generation of different intended actions.

The MTRNN shown in Fig. 5(a) consists of 3 subnetworks (slow, intermediate, fast dynamics networks; note that the numbers of levels can vary depending on application) each consisting of leaky integrator neural units that are assigned different time constants. The activation dynamics of a leaky integrator neuron can be described as a differential equation:

$$\tau \dot{u}^i = -u^i + \sum_j a^j w_{ij} \quad (3a)$$

$$a^i = 1/(1+e^{-u^i}) \quad (3b)$$

where τ represents the time constant of the neuron. When τ is set with a larger value, the time development of the neural activation becomes slower. With a smaller value, it becomes faster. Eq.3 is integrated over time using the difference method. The fast dynamics network in the lower level consists of two modular RNNs, one for predicting the proprioceptive state in the next step from current joint angle information, and the other for predicting low dimensional visual features in the next time step from current visual information.

During these humanoid robot learning experiments, the MTRNN was trained to generate a set of different visuo-proprioceptive trajectories corresponding to supervised targets by optimizing connectivity weights as well as the intention state corresponding to each trajectory. The intention state here is analogous to the PB value in the RNNPB, and corresponds with the initial states of neural units in the slow dynamics network of the MTRNN (see Fig. 5(a)). When learning begins, for each training sequence the initial state of the intention units is set to a small random value. The forward top-down dynamics initiated with this temporarily set initial state generates a predictive sequence. The error generated between the training sequence and the output sequence is back-propagated along the bottom-up path through the subnetworks with fast and moderate dynamics to the subnetwork with slow dynamics. This back-propagation is iterated backward through time steps via recurrent connections, whereby the connection weights within and between these subnetworks are modified in the direction of minimizing the error signal (at each time step). The error signal is also back-propagated through time steps to the initial state of the intention units, where these initial state values are modified.

Here, we see that learning proceeds through dense interactions between the top-down regeneration of training sequences and the bottom-up regression through these sequences by way of error signals, just as in the RNNPB. And as a result of this interaction, the robot learns a set of behavior primitive patterns such as reaching for an object, lifting the object up and down, or moving it left and right. These develop as distributed activation patterns in fast and intermediate dynamics networks while various control sequences for manipulating these primitive constructs

develop in the slow dynamics network (according to its initial sensitivity characteristics, see Fig. 5 (b)).

What explains the success of these models in performing such complex cognitive tasks? In the MTRNN, neural activity output from the higher level plays the role of bifurcation parameter for the lower level, like the PB vector in the RNNPB. Building from this work, Yamashita and Tani concluded that the decomposition of complex visuo-proprioceptive sequences into sequences of reusable primitives is achieved within this functional hierarchy due to subnetwork timescale differences.[22] Further experiments by Nishimoto and Tani and Arie and colleagues showed that MTRNNs can not only generate actual movements, but also diverse mental simulations of various intention states by performing closed-loop look ahead (so-called "off-line") prediction.[23] So, the question now becomes how to understand such functional hierarchies.

The development of functional hierarchies is captured in a well-known concept central to the study of complex adaptive systems, "downward causation," the causal relationship from global to local parts of a system.[24] A functional hierarchy develops by means of upward causation in terms of collective neural activity, both in forward activation dynamics and in error back-propagation. In the other direction, this development is subject to downward causation in terms of timescale difference, network topology, and environmental interaction. Note that these are strictly deterministic features of the system. Target conditions are determined. Current states are determined, and thereby optimal sequences of action are inferred. Next, we will look at an effort to articulate these temporal dynamics nondeterministically, in Friston's Bayesian predictive coding scheme formulated according to the free energy minimization principle.

3. THE FREE ENERGY MINIMIZATION PRINCIPLE

From the subjective perspective of an agent in the world, phenomena may be better described probabilistically than deterministically. Where upcoming anticipated optimal conditions are not pre-determined or perhaps even pre-determinable, the aforementioned models by Tani and colleagues can be extended into the probabilistic domain, as Friston has done.[25] Friston's main idea is to predict the next time step's perceptual states in terms both of their averages and their variances (or estimated accuracy). The average is a value arrived at according to prior instances, and actions undertaken on the basis of averages succeed best when deviations from the average are minimal. Variance, on the other hand, is a measure of the amount of difference between instances, and so can represent the accuracy of a prediction. Specifically, it can represent the estimated accuracy of a prediction, as a form of second-order prediction.[26]

Now, the exact formula for representing this idea is derived from the principle of free energy maximization.[27] Free energy can be computed by the addition of Gibbs energy G and Entropy E:

$$F = G + E \qquad (4\text{-a})$$

Then, F can be written in the following form.

$$F = \int q(z) ln P_\theta(x, z) dz - \int q(z) ln q(z) dz$$

Where $q(z)$ represents the prior distribution of the intention state, $P_\theta(x, z)$ represents joing probability distribution of observation x and the intention state Z parameterized by parameter θ. Then, free energy F can be transformed as:

$$\begin{aligned} F &= \int q(z) ln P_\theta(x, z) dz - \int q(z) ln q(z) dz \\ &= E_q[lnP(X, Z; \theta)] - E_q[lnQ(z)] \\ &= E_q[lnP(X, Z; \theta) - lnP(Z|X; \theta) + lnP(Z|X; \theta) - lnQ(z)] \\ &= E_q[lnP(X; \theta)] - KL[Q(Z)||P(Z|X; \theta)] \end{aligned} \qquad (4\text{-b})$$

The last form obtained in (4-b) is equal to the lower bound, L which is well known in the machine learning field. The first term represents the likelihood of reconstructing X by parameter θ and the second term represents minus KL divergence between the prior probability distribution of the intention state and the posterior distribution after observation of X with parameter θ. It can be seen that maximizing the free energy is equal to maximization of the lower bound. This lower bound L can be rewritten as:

$$L = \sum_s \sum_t \sum_i -1/2(\ln(v_{s,i}) + \frac{(\overline{o_{s,t,i}} - o_{s,t,i})^2}{v_{s,i}}) + \sum_s \sum_i -1/2(\ln(v^{IS}) + \frac{(IS_{s,i})^2}{\delta_{IS}{}^2}) \qquad (4\text{-c})$$

where $o_{s,t,i}$ is the ith dimension of the prediction output at time step t in the sth sequence, $\overline{o_{s,t,i}}$ is its teaching target, and v_{si} is its estimated variance, IS_{si} is the ith dimension of the intention state for the sth sequence, and δ_{IS} is its predefined deviation.

The generation, recognition, and learning of complex action sequences are possible through the maximization of free energy in the probabilistic domain just as the minimization of error performs similarly in the deterministic domain. According to the first term on the right-hand side of equation (4-c), the likelihood part can be maximized if variance is taken to be large even if the prediction square error is large. In this case, the agent has no reliable guide to anticipated future situations, so it simply relaxes any expectation of oncoming events. This would correspond with a reactive posture in a biological consciousness, for example. On the other hand, the likelihood might be small even though the prediction square error is small if the estimated variance is smaller than reality. In this case, an agent acts from intentions as if ends are predetermined, e.g., as if he has plotted all the necessary dimensions and their internal deviations so that action is facilitated and success presumed guaranteed. But, the agent ends up wrong about this, and suffers the correction. In human experience, having failed to adequately account for the world while having proceeded with laid plans in confidence is called "surprise." Similarly, according to Friston's free energy maximization principle (FEMP), the prediction square error divided by estimated variance represents the degree of surprise with interesting implications for inquiry into consciousness. For one thing, the measure of surprise may correlate with a measure of consciousness as the top-down accommodation of perceptual inputs at each time step.

According to the second term on the right-hand side of Eq. (4-c), the distance from the prior to the posterior can be minimized when the intention state of each sequence is distributed by following the Gaussian distribution with the predefined deviation δ_{IS}. The recognition of the intention in FEMP is to infer the optimal probabilistic distribution of the intention state for a given target sequence, maximizing free energy rather than infer a single optimal value minimizing the prediction error as in the RNNPB. Instantiating such a process in a model dynamic system is subject of the next section.

3.1 THE STOCHASTIC MTRNN MODEL

Because the original FEMP by Friston[28] was not implemented in any trainable neural network models, it was not clear how maximizing free energy in Eq.(4-a) might lead to successful learning of internal predictive models extracted from perceptual sequence data experienced in reality. For this reason, Murata and colleagues proposed a novel dynamic neural network, referred to as the stochastic-MTRNN (S-MTRNN) model.[29] This model incorporates Friston's (2010) FEMP into the deterministic learning model described in the last section, the MTRNN. The S- extends the original as it earns to predict subsequent inputs taking into account not only their **means** but also their "variances," or range of anticipated values. This means that if some segments of input sequences are more variable than others, then the time-dependent variances over these periods become larger. On the other hand, if some parts are less fluctuated, their variances are smaller. In effect, then, the S-MTRNN predicts the predictability of its own prediction for each dimension of the input sequences in a time-dependent manner. When variances are estimated as zero, then the S-MTRNN becomes a deterministic dynamic system like the original MTRNN, i.e., it anticipates zero variance. Therefore, it can be said that—depending on context—S-MTRNNs can develop either deterministic or stochastic dynamics, at which point arises the notion of probability and so some valuation of possible future states accordingly.

The model operates by means of maximizing the free energy described in Eq. (4) in all phases of learning, recognizing, and generating perceptual sequence patterns. An important development in the current model is that $v_{s,i}$ as estimated variance in the likelihood part of Eq. (4-c) is changed to a time variable valuable $v_{s,t,i}$ because its estimate can change at each time step of a perceptual sequence. The likelihood part exists to minimize the square error divided by estimated variance at each step. This means that the prediction error at a particular time step is pressured to be minimized more strongly when its estimated variance is smaller. Otherwise, the pressure to minimize prediction error is less.

Another development is that the intention state $IS_{s,i}$ in the part of KL divergence between the prior probability distribution of the intention state and the posterior distribution in Eq. (4-c) is now represented by the initial states of context units in all levels. The KL divergence part of Eq. (4-c) puts specific probabilistic distribution constraints on optimal initial states for all sequences with the parameter σ_{IS}^2. If σ_{IS}^2 is set with a large value, the distribution of initial states becomes wide. Otherwise, it becomes tight. By maximizing the free energy during the learning process, optimal connectivity weights for all teaching sequences, the probability distribution of the initial state for each teaching target sequence,[30] and the estimates of time-dependent variance for each sequence are obtained.

Figure 6 (a) shows the architecture of the S-MTRNN. The difference from the original MTRNN is that the S-MTRNN contains output units for predicting variances for all sensory dimensions at each time step.

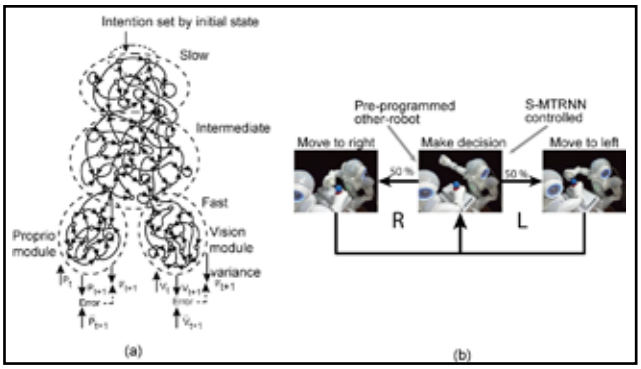

Figure 6. S-MTRNN model and the robotic experiment with the model. (a) The S-MTRNN contains additional output units for predicting variances for all sensory dimensions at each time step. (b) The "self-robot" learns to generate cooperative behaviors with the "other-robot."[31]

The next section reviews how the S-MTRNN performs in a particular robot task in the probabilistic domain.

3.2 LEARNING TO COOPERATE WITH OTHERS

A robotic experiment was conducted utilizing the S-MTRNN described in the preceding section (see Fig. 6).[32] The objective of this robot experiment was to examine how one robot can generate "cooperative" behavior by adapting to another robot's behavior, even though its predictions occasionally fail. The experiment used two "NAO" humanoid robots. One NAO robot, the "self-robot," attempted to generate cooperative behaviors with the "other" NAO robot. The other-robot's behavior was pre-programmed. The self-robot was controlled by the S-MTRNN model.

During the experiment, the other-robot repeated movement patterns and the self-robot was tutored to generate corresponding "cooperative" behaviors as it perceived the other's object movements. In order to do this, the self-robot needed to proactively initiate its own arm movement before sensing the actual movement initiated by the other-robot. The self-robot acquired this cooperative behavior skill through direct tutoring from the experimenter.[33] The self-robot observed the other-robot perform sequences of five movements, moving a colored object either to the left or to the right in all possible combinations (2^5 sequences). Then, the self-robot was required to generate cooperative behaviors by simultaneously moving its arm in the same direction as the other-robot. As it generated movements and adjusted to the other-robot's movements, differences emerged in the dynamics involved in predicting as well as

generating behavior between the two conditions of the wide and narrow initial states. The following explains how we tested these results in greater detail.

During the test phase of the experiment, the S-MTRNN was trained with 2^5 visuo-proprioceptive (VP) sequences during the tutoring process. This training was repeated twice, once with a small value for σ_{IS}^2 and then again with a large value in order to generate a narrow initial state distribution (Narrow-IS) and a wide initial state distribution (Wide-IS), respectively. Other-robot object movement (either to the left or to the right) was randomly determined from amongst the same 2^5 sequences so that the self-robot (S-MTRNN) would be unable to predict next movements reliably.

After training, closed-loop generation of "mental" imagery was performed for both wide and narrow training cases (i.e., offline rehearsal). During closed-loop operation, Gaussian noise corresponding to the estimated variance at each step was applied to the feedback from the previous step prediction output, and was input to the current step (see Figure 7 (a)).

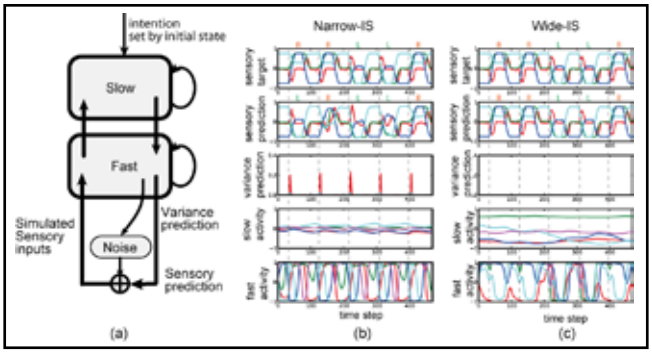

Figure 7. Generation of mental imagery via closed loop. (a) Closed-loop generation by S-MTRNN, generated sequences (b) in Narrow-IS case and (c) in Wide-IS case.[34]

In this way, mental imagery increasingly fluctuates as uncertainty of a prediction, i.e., the estimated variance, increases. In the example pictured in Fig. 7, the initial states were set with the values obtained upon learning the "RRLLR" trial sequence as performed by the other-robot. Fig. 7 (b) and (c) illustrate mental imagery in terms of prediction of VP sequences associated with estimated variance and internal neural activities in the fast and the slow subnetworks as generated by the S-MTRNNs trained under both Narrow-IS or Wide-IS conditions. In the Narrow-IS case, diverse decision sequences were generated even though all trials began from the same initial state. As the figure shows, estimated variance sharply peaks at decision points, but remains almost zero at other time steps. This implies that during training the S-MTRNN develops action primitives for moving left or right as two distinct chunks, and employs a probabilistic switching mechanism at decision points.

On the other hand in the Wide-IS case, the same decision sequence was repeatedly generated for the same given initial state. Fig. 7 (c) shows that the VP sequence for "RRLLR" was generated which seemed to be mostly the same as the target VP sequence. Here, it is important to note that the variance is estimated as almost zero for all steps including at decision points. This implies that mental imagery is generated as a deterministic predictive dynamics in the Wide-IS condition. Interestingly, for more than 20 branching instances before finally converging to cyclic branching, the robots'"mental imagery" (predictive dynamics) of next-movements was generated pseudo-randomly by means of transient chaos that developed in the slow dynamics part in the model network. This result is analogous to that of Namikawa et al., where complete chaos (with a positive Lyapunov exponent) instead of transient chaos appeared in the neural dynamics of an MTRNN.[35]

In the end, neural activity internal to the Narrow-IS and Wide-IS systems was quite different. In the Narrow-IS case, the neural activities in both the slow and the fast subnetworks showed the same values at all decision points. In the Wide-IS case, slow and fast neurons exhibited different activation patterns at each decision point through which the system was able to attempt to predict the subsequent move, left or right. There appears to be no such bias in activity at decision points in the Narrow-IS case, whereas there are top-down predictive biases imposed by specific top-level neural activation patterns at decision points in the Wide-IS case.

Let's look more closely at how the self-robot interacted with the other robot using the network trained in these two conditions, Wide-IS and Narrow-IS. Starting with arbitrary initial states, S-MTRNN generated one-step predictions for subsequent VP states upon perceiving current visual states via open-loop generation, while the other robot randomly moved a colored object sequences of five (see Figure 8 (a)).

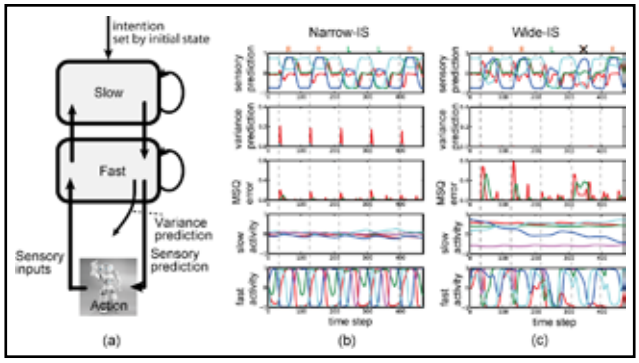

Figure 8. The results of the self-robot interacting with the other-robot by open-loop generation. (a) The scheme of the open-loop generation, (b) a sequence generated by the network trained with the Narrow-IS condition and (c) with the Wide-IS condition.[36]

Fig. 8 (b) and (c) show the results of open loop processing with the self-robot reacting to the other-robot as it generated the "RRLR" sequence for the Narrow-IS and the Wide-IS cases, respectively. Here, we observe that one-step prediction of VP states in the Narrow-IS case is quite successful, generating only a small error at each decision point. In contrast, one-step prediction in the Wide-IS case is much worse. In fact, the prediction error is significantly large at many decision points. Interestingly, at this juncture of the trials, the movement of the self-robot became erratic.

For example, in the fourth decision as illustrated in Fig. 8 (c), the self-robot moved its arm in the direction opposite to that of the other robot. And, although the self-robot seemed to try to follow the movements of the other-robot, its movements were significantly delayed.

The difference observed between the Wide-IS and Narrow-IS cases is best understood in terms of the different neural dynamic structures developed in these cases. In the case of the probabilistic dynamic structure developed in the Narrow-IS case, the behavior of moving either to the left or to the right is determined simply by following the other-robot by means of sensory reflex without any top-down bias.[37] In contrast, in the Wide-IS case, the top-down bias of internal neural activity at decision points is too strong to be modified by sensory input and incorrect movements are initiated and carried through.

3.3 INTRODUCING BOTTOM-UP ERROR REGRESSION

Next, consider an experiment that examines the effects of introducing an additional mechanism of bottom-up error regression into the learned neural dynamics during the course of behavior generation. This is a modified model which maximizes the likelihood LHreg for the time window of the immediate past by modifying the neural activation profile in this past window while fixing the connectivity weights (Fig. 9 (a)) as shown in Eq. (5).

$$LH_{reg} = \sum_{t'=t-W}^{t} \sum_{i} -1/2(\ln(v_{s,t,i}) + \frac{(\overline{o_{s,t,i}} - o_{s,t,i})^2}{v_{s,t,i}}) \quad (5)$$

where the time window is defined from t-W to t at the current time step and the activation states of the slow units at time step t-W (which is the onset of the window) are updated by back propagating the error signal generated. This error regression in terms of updating the activation state at the onset of the window and forward through the window is iterated multiple epochs during each time step in behavior generation. Again, as shown in Eq. 5, error back-propagates more strongly when the estimated variance (as the square error divided by the variance) is smaller.[38] An intuitive explanation is that in this scheme the internal representation in the immediate past window is rewritten for the sake of maximizing the likelihood for the on-going perception.

Fig. 9 (b) shows an example of developments during on-line behavior generation in the trained Wide-IS network using the present error regression scheme.

Clearly, neural activity in the gray area changes in a discontinuous manner with the generation of a sharp peak in prediction error only upon encountering unpredicted action by the other even though this error was rapidly reduced. Note that this sharp peak in prediction error is larger than that generated during on-line prediction in the case of the Narrow-IS as shown in Fig. 8 (b). In short, modulating higher-level neural activity by using error regression caused drastic changes in lower-level network activity including sensory predictions, and in this way prediction errors were rapidly minimized. Ultimately thus, the self-robot was able to re-situate its behavioral context

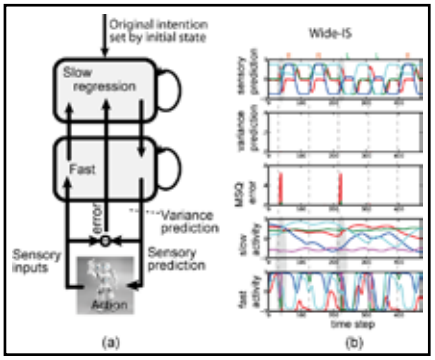

Figure 9. The results of on-line interaction using the error regression mechanism. (a) The on-line error regression scheme, (b) a sequence generated by the network trained with the Narrow-IS condition.[39]

immediately after encountering unpredictable events through dense interactions between top-down intentional prediction and bottom-up recognition of actual results.

How can we interpret these experimental results? First, let us summarize what we have just seen. In the Narrow-IS condition, probabilistic network dynamics develop generating actions in a sensory reflex manner. In contrast, proactive behaviors pursuant from deterministic predictions of next actions develop from the Wide-IS condition. It can be said that the Narrow-IS condition develops only weak top-down prior states while the Wide-IS condition develops strong top-down prior states. During the interaction of the self robot with the other robot, the self robot trained under the Narrow-IS condition could easily follow the action sequences arbitrarily determined by the other robot because it simply reacted to sensory inputs, with neural activity at decision points. On the other hand in the Wide-IS condition, the self-robot could not follow the action sequences of the other robot according to sensory inputs, because the top-down bias originating from the initial state was too strong. However, when the error regression scheme was applied utilizing the prediction error generated, the actions of the Wide-IS self robot were modified immediately by means of rapid changes in internal neural states. This bottom-up modulation can be quite strong because the variance is estimated as small in the case of the Wide-IS. This is due to the development of a deterministic dynamic structure, one that plans its next action, and that can be said to "have" a future toward which it has effectively committed itself through proactive cognitive agency given strong top-down prior states. On the other hand, the same force is not so strong in the probabilistic (Narrow-IS) case because the estimated variance at decision points is large. This is to say that the Narrow-IS has plotted no future condition beyond immediate reaction, and has thus cannot be said to "have" a future in this same way.

Consider these Wide and Narrow conditions from the Bayesian viewpoint. In the Bayesian framework, the S-MTRNN represents a likelihood function which maps intention state to a probability distribution of up-coming perceptual states. In these experiments, the distribution of intention states (initial states) was constrained by either the Wide distribution or the Narrow distribution, and the experiments show that the Wide distribution of intention states develops a deterministic dynamics with strong top-down prior states, whereas the Narrow distribution develops a probabilistic process which is a purely reactive process.

3.4 TIME PERCEPTION BY "EMBODIED" RNNS

Now, we come back the main issue of consciousness. This section briefly looks at the problem of time perception in light of Francisco Varela's "present-time consciousness."[40]

Tani and Nolfi postulated that "consciousness" arises at the very moment of segmenting the perceptual flow by means of error regression.[41] Varela's "present-time consciousness" arises similarly.[42] First, Varela considered that the immediate past does not belong to a representational conscious memory, but just to an impression consistent with Husserl's idea of retention.[43] So, his question was how the immediate past, experienced just as an impression, could slip into a distant past which can be retrieved through a conscious memory operation later on. And, in response, he proposed that nonlinear dynamics theory could be used as the formal descriptive tool for this phenomenon. By using the phenomenon of the spontaneous flipping of a Necker cube as an example, he explained that the dynamic properties of intermittent chaos characterized by its spontaneous shifts between static and rapid transition modes could explain the paradox of continuous, yet also segmented, time perception.

On his consideration, we may still ask how such spontaneous shifts as those realized by intermittent chaos can be linked to conscious experience. Although Thompson and Varela explain that such shifts are accompanied by shifts in neuronal bias, what is the formal mechanism of this process?[44] Tani proposes that consciousness arises in the correction and modification of dynamic structures which, in biological cognition, are generated in higher cortical areas.[45] The following attempts to account for the development of levels of conscious experience in terms of the development of the predictive RNN models described so far in the current paper.

In subjective terms, firstly an agent experiences a continuous perceptual flow without this flow being articulated in any way, that is without this flow representing any discernible thing or event. However, there should be retention and protention in this primordial level, as explained by Husserl (see the last footnote).[46] Retention and protention are used to designate the experienced sense of the immediate past and the immediate future. They are a part of automatic processes and cannot be controlled consciously. Husserl believed that the subjective experience of "nowness" is extended to include fringes both in the experienced sense of the past and the future in terms of retention and protention. This description of retention and protention at the so-called "pre-empirical"level by Husserl seems to directly correspond to what the basic RNN (as illustrated in Fig. 1 in the earlier section) is performing. The RNN predicts its next state by retaining the past flow in a context dependent way as has been described. This self-organized contextual flow in the forward dynamics of RNNs could account for the phenomenon of retention, whereas prediction based on this contextual flow naturally corresponds to protention.

With Husserl's idea of "nowness" in terms of retention and protention, the following question arises: Where is the "nowness" bounded? Husserl and Varela believe that the immediate past does not belong to a representational conscious memory but just to an impression, as suggested above. This led Varela to wonder what kind of mechanism qualitatively changes an experience from just an impression to an episodic consciously retrievable event.[47] Husserl's goal was to explain the emergence of objective time from the pre-empirical level of retention and protention,[48] and he seems to think that the sense of objective time should emerge as a natural consequence of organizing experience into one consistent linear sequence. Still, the question remains: What is the underlying mechanism for this?

One way of approaching this question is to consider first that "nowness" can be bounded where the flow of experience is segmented. Imagine that "Re Fa La" and "Do Mi So" are frequently heard phrases. The sequential notes of "Do Mi So" constitute a chunk within the sound stimulus flow, because the sequence can be predicted perfectly by developing coherence between the predictive neural dynamics and the perceptual flow. Within the chunk, everything proceeds smoothly, automatically, and unconsciously. However, when we hear a next phrase of "Re Fa La" after "Do Mi So" (considering that this second phrase is not necessarily predictable from the first one) a temporal incoherence emerges as prediction error is generated in the transition between the two phrases. The central thesis here is that consciousness arises as the agent attempts to deal with the uncertainty or open possibility between the two.

In Tani and colleagues' RNN models, the winner module is switched from one to another in MixRNNs or PB is shifted in RNNPB by means of error regression when the external perceptual flow does not match with the internal flow of the prediction. This matching is primarily occurring in the window of the immediate past, as described above. When the prediction is betrayed, the perceptual flow is segmented into chunks associated with shifts of gates or PBs, minimizing prediction error. Those segmented chunks are no longer just parts of the flow, but events that are identified by an activated local module or a PB vector value, e.g., as one of the NAO robot's behavior primitives. Because of delays in the error minimization process for optimizing gate openings or PB vector, this identification process can be time consuming. This might explain the phenomenological observation that the flow of the immediate past is experienced only as an impression, which later becomes a consciously retrievable object after being segmented. This may correspond to an observation of postdiction evidenced in neuroscience.[49] See Figure 10 for an illustration of the idea.

The higher level RNN in MixRNNs, RNNPBs, or MTRNNs learns the sequences of the identified events and becomes able to regenerate them as a narrative.

During memory retrieval however, the perceptual flow can be reconstructed only in an indirect way since the flow is now represented by combining a set of commonly used behavior primitives. Although such reconstructions provide for compositionality as well as generalization in representing perceptual flow, they might lose subtle differences or uniqueness in each instance of experience depending on the capacities to retain perceptual dynamics.

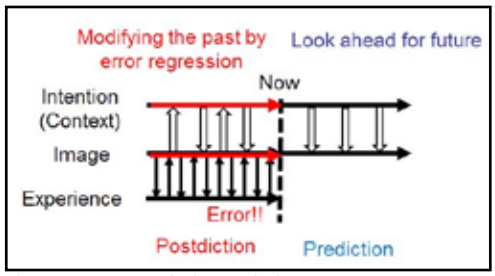

Figure 10. Prediction of future based on the postdiction of the past.

Consequently, we presume that the sense of objective time appears when experience of the perceptual flow is reconstructed as a narrative in a compositional form, while losing its peculiarity.

From the Bayesian perspective of Friston's FEMP, the agent becomes able to reflect on the episodic sequence with self-estimated certainty when the Narrow-IS condition is applied to S-MTRNN, as shown in the aforementioned experiments by Murata and colleagues.[50] At this stage, the agent finally becomes able to represent its own episodic sequence in terms of a probabilistic model by inferring that each chunk (moving left or right) simply arises with a certain probability. This is a crucial transition from first reflecting on its own experience as a deterministic "one time only" episodic sequence occurring only in that way, to viewing it as a probability which could have taken place in other ways. From the latter point of view, the agent is successful in ultimately objectifying its own experience by reconstructing it into a generalized model accounting for possible interactions between its self and others. However, it is interesting to note that the agent in this stage does not maintain anymore the subjectivity of naively intending for an uncertain future, because all it maintains is ultimately objectified models of probable futures. Together, these stages of development should begin to account for the process of an agent attaining a reflective self which is only then potentially maintained, for example through inner discourse and conscious narration and which only then results in truly direct subjective experience, the characteristic "mineness" of h-consciousness as revealed in our last paper.

3.5 DISCUSSION

With the composition of intentional sequences, we may understand surprise as their unexpected correction resulting in consciousness. To this, one may object that one becomes conscious of many things without surprise, but this objection is easily answered. Let us consider that intentional processes drive the whole neural network dynamics including the peripheral subnetworks by means of chaos or transient chaos developed in the higher cognitive brain area in order to act on the world in achieving some end of agency such as in Murata's robot experiment and in Namikawa et al.[51] At this moment of acting, some prediction errors may be generated at the very least because the world is inevitably unpredictable due to its openness relative for instance human cognitive agency. Then, at the very moment when the intention state is modulated by those errors back-propagated from the peripheral to the higher level, the agent becomes conscious of the formulation of intention upon which it has acted and only in a "postdictive" manner,[52] i.e., when the intention in the past window is rewritten for the sake of accounting for the current perception, there is consciousness.

With this, we may ask if we can apply the aforementioned analysis to account for the delayed awareness of "free will" as for example evidenced in the famous Libet experiments?[53] One might imagine that no prediction errors are to be associated with decisions about pressing a button as in Libet's experiments. However, in order to initiate a particular movement, internal neural activity in peripheral areas including muscle potential states must prepare for action. With this in mind, prediction errors may arise when the higher cognitive level such as the prefrontal cortex (PFC) or supplementary motor area (SMA) suddenly attempts to drive the lower peripheral processes such as the motor area and somatosensory area through the parietal area, possibly by chaos, to generate a specific movement when the lower parts are not yet prepared for it (see Figure 11).

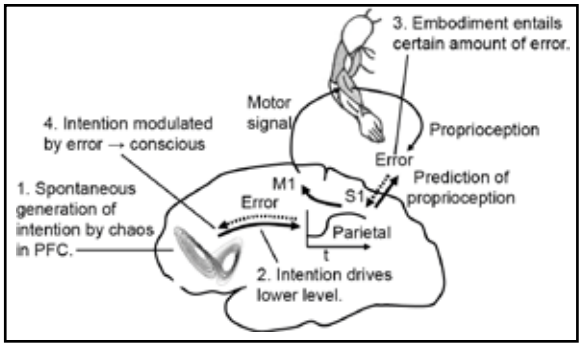

Figure 11. Explanation of how free will can be initiated unconsciously and how one can become consciously aware of it with delay.[54]

In such a situation, a gap may appear between the higher level with the sudden urge for generating the movement and lower level processes which are not yet ready for it. This gap appears in the system as a sort of prediction error, with the intention to act confounded by factors internal to the system as a whole but still external to the intentional processes, themselves. This difference, then, between what is ideally intended and its practical exercise may then cause the conscious awareness of one's own intention, again with a delay as described by Libet, Gleason and Wright,[55] and as having been conjectured by Tani.[56] To sum up, we consider that free will may originate unconsciously by means of the cortical deterministic chaos which can become an object of conscious awareness only after a certain delay under embodiment constraint in terms of postdiction.[57]

Before concluding the current paper, give a close look at the process of error regression at the moment that prediction error increases due to unexpected perception. This error regression process involves the nontrivial phenomena of circular causality, analysis of which reveals subtle characteristics of the conscious process. In simple situations such as shown in the experiments by Murata and

colleagues wherein possible actional decisions are only two, either moving an arm to the left or to the right, the conflictive situation can be resolved instantly by sudden modulation of the intention by error regression.[58] However, realistic situations are more complex, for example, when a system has to perform online modification of a goal-directed plan by searching among various possible combinations of behavior primitives by means of error regression, while retaining immediate integrity in the face of environmental forces, i.e., adapting to rapid changes of the current situation until the newly formulated intention can be carried forward.

4. CONCLUSION

The current paper reviewed a series of neurorobotics studies conducted by Jun Tani and colleagues that attempt to provide a purely formal, structural account of dynamical processes essential for consciousness. The core ingredients of Tani's models are prediction and postdiction through predictive coding and implemented in different recurrent neural network (RNN) models that together represent a progression from reflexive to proactive, self-reflective and creative agency. The review moved from simple to more complex model hierarchies.

Robotics experiments employing these models clarified dynamics inherent in levels of consciousness from momentary self-consciousness (surprise) to narrative self and reflective self-consciousness (the "chunking" of experience and the articulation of perceptual flow according to developing action potentials). The paper concluded with a brief phenomenological analysis of time perception within this family of models, including model extensions accounting for free will and its characteristic postdictive conscious awareness. In the next paper, we will begin with some of Tani and colleagues' work on these model extensions into more complex situations, before returning to Boltuc's naturalistic non-reductionism and a philosophical analysis of any claim to consciousness of artificial systems.

NOTES

1. The correspondence should be sent to tani1216jp@gmail.com
2. Here it is interesting to note that predictive coding is inspired by studies on biological systems, so computational architectures employing predictive coding are by definition instances of a biological approach. R. N. Rao and D. H. Ballard, "Predictive Coding in the VIsual Cortex: A Functional Interpretation of Some Extra-Classical Receptive-Field Effects," *Nature Neuroscience* 2 (1999): 79–87; J. Tani and S. Nolfi, "Learning to Perceive the World as Articulated: An Approach for Hierarchical Learning in Sensory-Motor Systems," *Neural Networks* 12, no. 7 (1999): 1131–41; K. Friston, "A Theory of Cortical Responses," *Philosophical Transactions of the Royal Society B: Biological Sciences* 360, no. 1456 (2005): 815–36.
3. J. Tani, "Autonomy of 'Self' at Criticality: The Perspective from Synthetic Neuro-Robotics," *Adaptive Behavior* 17, no. 5 (2009): 421–43; A. Clark, *Surfing Uncertainty: Prediction, Action, and the Embodied Mind* (NY: Oxford University Press, 2015); J. Tani, *Exploring Robotic Minds: Actions, Symbols, and Consciousness as Self-Organizing Dynamic Phenomena* (New York: Oxford University Press, 2016).
4. J. Tani, "Learning to Generate Articulated Behavior Through the Bottom-Up and the Top-Down Interaction Process," *Neural Networks* 16 (2003): 11–23.
5. Friston, "A Theory of Cortical Responses"; K. Friston, "The Free-Energy Principle: A Unified Brain Theory?" *Nature Reviews Neuroscience* 11 (2010): 127–38.
6. M. I. Jordan, "Serial Order: A Parallel Distributed Processing Approach," Technical Report, California University, San Diego, 1986; J. L. Elman, "Finding Structure in Time," *Cognitive Science* 14 (1990): 179–211; R. J. Williams and D. Zipser, "Finding Structure in Time," Institute for Cognitive Science Report, University of California, San Diego, 1990.
7. Ibid.
8. C.f. J. Tani and N. Fukumura, "Embedding a Grammatical Description in Deterministic Chaos: An Experiment in Recurrent Neural Learning," *Biological Cybernetics* 72, no. 4 (1995): 365–70.
9. D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Internal Representations By Error Propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, ed. D. E. Rumelhart and J. L. Mclelland (Cambridge, MA: The MIT Press, 1986).
10. J. Tani and S. Nolfi, "Self-Organization of Modules and Their Hierarchy in Robot Learning Problems: A Dynamical Systems Approach," *News Letter on System Analysis for Higher Brain Function Research Project* 2, no. 4 (1997): 1–11; Tani and Nolfi, "Learning to Perceive the World as Articulated: An Approach for Hierarchical Learning in Sensory-Motor Systems."
11. Tani, "Learning to Generate Articulated Behavior Through the Bottom-Up and the Top-Down Interaction Process"; J. Tani and M. Ito, "Self-Organization of Behavioral Primitives as Multiple Attractor Dynamics: A Robot Experiment," *Systems, Man, and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 33, no. 4 (2003): 481–88.
12. Rizzolatti et al. ("Mirror Neuron: A Neurological Approach to Empathy," in *Neurobiology of Human Values* [Springer-Verlag Berlin Heidelberg, 1995]) as explored in J. Tani, M. Ito, and Y. Sugita, "Self-Organization of Distributedly Represented Multiple Behavior Schemata in a Mirror System: Reviews of Robot Experiments Using RNNPB," *Neural Networks* 17 (2004): 1273–89.
13. Tani and Nolfi, "Self-Organization of Modules and Their Hierarchy in Robot Learning Problems"; Tani and Nolfi, "Learning to Perceive the World as Articulated."
14. Here, recall that it is the object of the network to minimize error through the error back-propagation through time (BPTT) algorithm.
15. Tani and Nolfi, "Learning to Perceive the World as Articulated."
16. Tani, "Learning to Generate Articulated Behavior Through the Bottom-Up and the Top-Down Interaction Process"; Tani and Ito, "Self-Organization of Behavioral Primitives as Multiple Attractor Dynamics: A Robot Experiment."
17. Tani and Nolfi, "Self-Organization of Modules and Their Hierarchy in Robot Learning Problems"; Tani and Nolfi, "Learning to Perceive the World as Articulated"; and Tani, "Learning to Generate Articulated Behavior Through the Bottom-Up and the Top-Down Interaction Process."
18. It is interesting to note here a formal similarity with "abductive" agent-level evolutionary models.
19. Tani and Nolfi, "Self-Organization of Modules and Their Hierarchy in Robot Learning Problems"; Tani and Nolfi, "Learning to Perceive the World as Articulated."
20. Tani, "Learning to Generate Articulated Behavior Through the Bottom-Up and the Top-Down Interaction Process."
21. Y. Yamashita and J. Tani, "Emergence of Functional Hierarchy in a Multiple Timescale Neural Network Model: A Humanoid Robot Experiment," *PLoS Computational Biology* 4, no. 11 (2008): e1000220.
22. Ibid.
23. R. Nishimoto and J. Tani, "Development of Hierarchical Structures for Actions and Motor Imagery: A Constructivist View from Synthetic Neuro-Robotics Study," *Psychological Research* 73, no. 4 (2009): 545–58; and H. Arie, T. Endo, T. Arakaki, S. Sugano, and J. Tani, "Creating Novel Goal-Directed Actions at Criticality: A Neuro-robotic Experiment," *New Mathematics and Natural Computation* 5, no. 01 (2009): 307–34.

24. D. T. Campbell, "'Downward Causation' in Hierarchically Organized Biological Systems," in *Studies in the Philosophy of Biology* (Macmillan Education UK, 1974), 179–86; E. Thompson and F. J. Varela, "Radical Embodiment: Neural Dynamics and Consciousness," *Trends in Cognitive Sciences* 5, no. 10 (2001): 418–25.

25. Friston, "A Theory of Cortical Responses"; K. Friston, "Hierarchical Models in the Brain," *PLoS Computational Biology* 4, no. 11 (2008): e1000211.

26. It is important here to bear in mind that these are systems enabling agency, and so an action that ends very far from a target is much worse than one which ends close enough. It is not as innocuous as simply getting something wrong. If variance is high, then a prediction which hits its target is extremely accurate, such that in the real world it may not be believed, e.g., "too good to be true." However, when variance is high, it also means that average values do not effectively inform action. Acting on the basis of an average will always in the long run result in error proportional to variance. Once this is understood, then an agent may apply estimated variance in the prediction of optimal next actions, as this value may inform the agent what to expect given prior instances, reducing error over the long run.

    This formal model recalls Plato's concern with the science of science that is ultimately knowledge of good and bad, a second-order understanding that for example directs sight but never sees a thing, c.f. Charmides.

27. Friston, "A Theory of Cortical Responses."

28. Friston, "The Free-Energy Principle: A Unified Brain Theory?"

29. S. Murata, Y. Yamashita, H. Arie, T. Ogata, S. Sugano, and J. Tani, "Learning to Perceive the World as Probabilistic or Deterministic via Interaction with Others: A Neuro-robotics Experiment," *IEEE Trans. on Neural Networks and Learning Systems*. 2015. doi:10.1109/TNNLS.2015.2492140

30. Recognition density in Friston, "The Free-Energy Principle: A Unified Brain Theory?"

31. Redrawn from Murata et al., "Learning to Perceive the World as Probabilistic or Deterministic via Interaction with Others."

32. Ibid.

33. C.f. Yamashita and Tani, "Emergence of Functional Hierarchy in a Multiple Timescale Neural Network Model: A Humanoid Robot Experiment"; J. Namikawa, R. Nishimoto, and J. Tani, "A Neurodynamic Account of Spontaneous Behavior," *PLoS Computational Biology* 7, no. 10 (2011): e1002221.

34. Redrawn from Murata et al., "Learning to Perceive the World as Probabilistic or Deterministic via Interaction with Others."

35. Namikawa et al., "A Neurodynamic Account of Spontaneous Behavior."

36. Redrawn from Murata et al., "Learning to Perceive the World as Probabilistic or Deterministic via Interaction with Others."

37. Recall that the slow and fast networks showed the same dynamics at each point.

38. In terms of human experience, it feels worse being wrong when sure that he/she is right than when it is a recognized matter of chance.

39. Redrawn from Murata et al., "Learning to Perceive the World as Probabilistic or Deterministic via Interaction with Others."

40. F. J. Varela, "Present-Time Consciousness," *Journal of Consciousness Studies* 6, nos. 2-3 (1999): 111–40.

41. Tani and Nolfi, "Learning to Perceive the World as Articulated."

42. Varela, "Present-Time Consciousness."

43. E. Husserl, "The Phenomenology of Internal Time Consciousness," trans. J. S. Churchill (Bloomington, IN: Indiana University Press, 1964). Husserl introduced the famous idea of "retention" and "protention" for explaining the paradoxical nature of "nowness." He used an example of hearing a sound phrase such as "Do Mi So" for explaining the idea. When we hear the note "Mi," we would still perceive a lingering impression of "Do," and at the same time we would anticipate hearing the next note of "So." The former is called retention and the latter protention. These terms are used to designate the experienced sense of the immediate past and the immediate future.

44. E. Thompson and F. J. Varela, "Radical Embodiment: Neural Dynamics and Consciousness," *Trends in Cognitive Sciences* 5, no. 10 (2001): 418–25.

45. J. Tani, "An Interpretation of the 'Self' from the Dynamical Systems Perspective: A Constructivist Approach," *Journal of Consciousness Studies* 5, nos. 5/6 (1998): 516–42; Tani and Nolfi, "Learning to Perceive the World as Articulated: An Approach for Hierarchical Learning in Sensory-Motor Systems"; Tani, "Learning to Generate Articulated Behavior Through the Bottom-Up and the Top-Down Interaction Process."

46. Husserl, "The Phenomenology of Internal Time Consciousness." See footnote 51.

47. Varela, "Present-Time Consciousness."

48. Husserl, "The Phenomenology of Internal Time Consciousness."

49. D. M. Eagleman and T. J. Sejnowski, "Motion Integration and Postdiction in Visual Awareness," *Science* 287, no. 5460 (2000): 2036–38; S. Shimojo, "Postdiction: Its Implications on Visual Awareness, Hindsight, and Sense of Agency," *Frontiers in Psychology* 5 (2014): 196.

50. Murata et al., "Learning to Perceive the World as Probabilistic or Deterministic via Interaction with Others."

51. Ibid. and in Namikawa et al., "A Neurodynamic Account of Spontaneous Behavior."

52. Shimojo, "Postdiction: Its Implications on Visual Awareness, Hindsight, and Sense of Agency."

53. C.f. B. Libet, E. W. Wright, and C. A. Gleason, "Preparation- or Intention-to-Act, in Relation to Pre-event Potentials Recorded at the Vertex," *Electroencephalography and Clinical Neurophysiology* 56, no. 4 (1983): 367–72; B. Libet, "Unconscious Cerebral Initiative and the Role of Conscious Will in Voluntary Action," *Behavioral and Brain Sciences* 8 (1985): 529–39.

54. Redrawn from J. Tani, *Exploring Robotic Minds: Actions, Symbols, and Consciousness as Self-Organizing Dynamic Phenomena* (New York: Oxford University Press, 2016) with permission from Oxford University Press.

55. Libet, Gleason, and Wright, "Preparation- or Intention-to-Act, in Relation to Pre-event Potentials Recorded at the Vertex." The preceding interpretation accords with recent work in the sense of self-agency as it changes due to feedback. In short, the more that an agent perceives itself to be in control of outcomes, the more it feels the sense of ownership of actions performed (c.f. N. Kumar, J. A. Manjaly, and K. P. Miyapuram, "Feedback about Action Performed Can Alter the Sense of Self-Agency," *Frontiers in Psychology*, February 25, 2014).

56. Tani, *Exploring Robotic Minds: Actions, Symbols, and Consciousness as Self-Organizing Dynamic Phenomena*.

57. Eagleman and Sejnowski, "Motion Integration and Postdiction in Visual Awareness"; Shimojo, "Postdiction: Its Implications on Visual Awareness, Hindsight, and Sense of Agency."

58. Murata et al., "Learning to Perceive the World as Probabilistic or Deterministic via Interaction with Others."

REFERENCES

Arie, H., T. Endo, T. Arakaki, S. Sugano, and J. Tani. "Creating Novel Goal-Directed Actions at Criticality: A Neuro-robotic Experiment." *New Mathematics and Natural Computation* 5, no. 01 (2009): 307–34.

Boltuc, P. "The Philosophical Issue in Machine Consciousness." *International Journal of Machine Consciousness* 1, no. 1 (2009): 155–76.

Campbell, D. T. "'Downward Causation' in Hierarchically Organized Biological Systems." In *Studies in the Philosophy of Biology*, 179–86. Macmillan Education UK, 1974.

Clark, A. *Surfing Uncertainty: Prediction, Action, and the Embodied Mind*. NY: Oxford University Press, 2015.

Eagleman, D. M., and T. J. Sejnowski. "Motion Integration and Postdiction in Visual Awareness." *Science* 287, no. 5460 (2000): 2036–38.

Elman, J. L. "Finding Structure in Time." *Cognitive Science* 14 (1990): 179–211.

Friston, K. "A Theory of Cortical Responses." *Philosophical Transactions of the Royal Society B: Biological Sciences* 360, no. 1456 (2005): 815–36.

———. "Hierarchical Models in the Brain." *PLoS Computational Biology* 4, no. 11 (2008): e1000211.

———. "The Free-Energy Principle: A Unified Brain Theory?" *Nature Reviews Neuroscience* 11 (2010): 127–38.

Husserl, E. "The Phenomenology of Internal Time Consciousness." Translated by J. S. Churchill. Bloomington, IN: Indiana University Press, 1964.

Jordan, M. I. "Serial Order: A Parallel Distributed Processing Approach." Technical Report. California University, San Diego, 1986.

Kumar, N., J. A. Manjaly, and K. P. Miyapuram. "Feedback about Action Performed Can Alter the Sense of Self-Agency." *Frontiers in Psychology*, February 25, 2014. doi:10.3389/fpsyg.2014.00145

Libet, B. "Unconscious Cerebral Initiative and the Role of Conscious Will in Voluntary Action." *Behavioral and Brain Sciences* 8 (1985): 529–39.

Libet, B., E. W. Wright, and C. A. Gleason. "Preparation- or Intention-to-Act, in Relation to Pre-event Potentials Recorded at the Vertex." *Electroencephalography and Clinical Neurophysiology* 56, no. 4 (1983): 367–72.

Murata, S., Y. Yamashita, H. Arie, T. Ogata, S. Sugano, and J. Tani. "Learning to Perceive the World as Probabilistic or Deterministic via Interaction with Others: A Neuro-robotics Experiment." *IEEE Trans. on Neural Networks and Learning Systems*. 2015. doi:10.1109/TNNLS.2015.2492140

Namikawa, J., R. Nishimoto, and J. Tani. "A Neurodynamic Account of Spontaneous Behavior." *PLoS Computational Biology* 7, no. 10 (2011): e1002221.

Nishimoto, R., and J. Tani. "Development of Hierarchical Structures for Actions and Motor Imagery: A Constructivist View from Synthetic Neuro-Robotics Study." *Psychological Research* 73, no. 4 (2009): 545–58.

Rao, R. N., and D. H. Ballard. "Predictive Coding in the VIsual Cortex: A Functional Interpretation of Some Extra-Classical Receptive-Field Effects." *Nature Neuroscience* 2 (1999): 79–87.

Reed, E. S., and D. Schoenherr. "The Neuropathology of Everyday Life: On the Nature and Significance of Microslips in Everyday Activities." Unpublished manuscript. 1992.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams. "Learning Internal Representations By Error Propagation." In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, edited by D. E. Rumelhart and J. L. Mclelland. Cambridge, MA: The MIT Press, 1986.

Shimojo, S. "Postdiction: Its Implications on Visual Awareness, Hindsight, and Sense of Agency." *Frontiers in Psychology* 5 (2014): 196. 10.3389/fpsyg.2014.00196.

Tani, J. "An Interpretation of the 'Self' from the Dynamical Systems Perspective: A Constructivist Approach." *Journal of Consciousness Studies* 5, nos. 5/6 (1998): 516–42.

———. "Learning to Generate Articulated Behavior Through the Bottom-Up and the Top-Down Interaction Process." *Neural Networks* 16 (2003): 11–23.

———. "The Dynamical Systems Accounts for Phenomenology of Immanent Time: An Interpretation By Revisiting a Robotics Synthetic Study." *Journal of Consciousness Studies* 11, no. 9 (2004): 5–24.

———. "Autonomy of 'Self' at Criticality: The Perspective from Synthetic Neuro-Robotics." *Adaptive Behavior* 17, no. 5 (2009): 421–43.

———. *Exploring Robotic Minds: Actions, Symbols, and Consciousness as Self-Organizing Dynamic Phenomena*. New York: Oxford University Press, 2016.

Tani, J., and N. Fukumura. "Embedding a Grammatical Description in Deterministic Chaos: An Experiment in Recurrent Neural Learning." *Biological Cybernetics* 72, no. 4 (1995): 365–70.

Tani, J., and S. Nolfi. "Self-Organization of Modules and Their Hierarchy in Robot Learning Problems: A Dynamical Systems Approach." *News Letter on System Analysis for Higher Brain Function Research Project* 2, no. 4 (1997): 1–11.

———. "Learning to Perceive the World as Articulated: An Approach for Hierarchical Learning in Sensory-Motor Systems." *Neural Networks* 12, no. 7 (1999): 1131–41.

Tani, J., and M. Ito. "Self-Organization of Behavioral Primitives as Multiple Attractor Dynamics: A Robot Experiment." *Systems, Man, and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 33, no. 4 (2003): 481–88.

Tani, J., M. Ito, and Y. Sugita. "Self-Organization of Distributedly Represented Multiple Behavior Schemata in a Mirror System: Reviews of Robot Experiments Using RNNPB." *Neural Networks* 17 (2004): 1273–89.

Thompson, E., and F. J. Varela. "Radical Embodiment: Neural Dynamics and Consciousness." *Trends in Cognitive Sciences* 5, no. 10 (2001): 418–25.

Varela, F. J. "Present-Time Consciousness." *Journal of Consciousness Studies* 6, nos. 2-3 (1999): 111–40.

Williams, R. J., and D. Zipser. "Finding Structure in Time." Institute for Cognitive Science Report. University of California, San Diego, 1990.

Yamamoto, J., J. Suh, D. Takeuchi, and S. Tonegawa. "Successful Execution of Working Memory Linked to Synchronized High-Frequency Gamma Oscillations." *Cell* 157, no. 4 (2014): 845–57.

Yamashita, Y., and J. Tani. "Emergence of Functional Hierarchy in a Multiple Timescale Neural Network Model: A Humanoid Robot Experiment." *PLoS Computational Biology* 4, no. 11 (2008): e1000220.

Kant on Constituted Mental Activity

Richard Evans
IMPERIAL COLLEGE, UK

1 INTRODUCTION

Consider the following functionalist claim:

> There is an architecture, describable in the language of computer science, such that any creature or machine that realises this architecture thereby counts as a cognitive agent, an agent with original (non-derivative) intentionality.

Some of the more practically minded among us will be dissatisfied with this existentially quantified assertion: rather than just saying that there is *some* such architecture, it would be much more helpful to know exactly what this architecture is. What sort of architecture could satisfy such a claim?

I believe the answer to this question has been hiding in plain sight for over two hundred years: in *The Critique of Pure Reason*, Kant provides a detailed description of just such an architecture.

At the heart of Kant's vision is the *self-legislating agent*: an agent who constructs rules that he then solemnly follows. The Kantian cognitive architecture is a particular type of *computational process*: a rule-induction process. If this rule-induction process satisfies certain constraints, then—Kant claims—the process' internal activities count as cognitive activities.

This paper sketches the philosophical background behind this architecture. It attempts to motivate and defend Kant's vision of a self-legislating computational agent.[1]

2 MENTAL ACTIVITY AS CONSTITUTED ACTIVITY

We are familiar with the idea that social activity is *constituted* activity. Pushing the wooden horse-shaped piece forward counts, in the right circumstances, as moving the knight to

king's bishop three. Jones' running away counts, in the right circumstances, as desertion. An utterance of the words "I do" counts, in the right circumstances, as an acceptance of marriage vows. These social actions are things we can only do *indirectly*, by doing something else. A social action is not something we can *just do*.

Kant's cardinal innovation, as I read him, is to see *mental* activity as constituted activity. This plurality of sensory perturbations counts, in the right circumstances, as my representing a red triangle. This activity of rule application counts, under the right circumstances, as my seeing an apple. This activity of rule construction counts, under the right circumstances, as my forming the belief that Caius is mortal. The surprising Kantian claim is that mental activity is itself constituted. We have to perform a certain type of ritual in order to experience a world at all.

2.1 FROM COUNTS-AS TO COUNTING-AS

Let us start by considering the *activity* of counting-as:

- Jones counts Smith's contortion of the lips as a delighted smile
- The sergeant counts Jones' running away as desertion
- The teacher counts the boy's squiggle as an "s"
- The vicar counts the utterance of the words "I do" as an acceptance of the marriage vows

Notice that these examples describe the *activity* of counting-as, rather than the mere *relation* of counts-as. The counts-as relation is commonly formulated as:

> x counts as y (in context c)

This sentence, ascribing a three-place relation between x, y and c, ignores the *person* who is doing the counting-as, and the *business of counting itself*, and focuses solely on the resulting judgment. If we want to acknowledge the individual performing the counting, and the activity of counting itself, we would write it as:

> agent *a* counts x as y (in context c)

This sentence describes the activity of counting-as, and makes explicit the person who is *doing* the counting.

Under what circumstances would it be ok to forget about the person doing the counting? Perhaps it would be ok to suppress the agent and the activity of counting-as in cases where everyone agreed about what counted-as what, where massive agreement in counting-as was taken for granted. Throughout the *Investigations*, Wittgenstein repeatedly asks us to stop taking this mass communal agreement for granted. He demands "what if one person reacts in one way and another in another?"[2] For example, he considers the case where:

> a person naturally reacted to the gesture of pointing with the hand by looking in the direction of the line from finger-tip to wrist, not from wrist to finger-tip.[3]

The divergence here is a difference in what activity the deviant person is counting the gesture as. The deviant is counting the gesture as pointing in the opposite direction from what "we"[4] count the gesture as.

Whenever Wittgenstein talks about counts-as, he is careful to talk about the activity of counting-as, rather than an abstract relation of counts-as that presupposes communal agreement:

> But now imagine a game of chess translated according to certain rules into a series of actions which we do not ordinarily associate with a game—say into yells and stamping of feet. ... *Should we still be inclined to count them as* playing a game? What right would one have to say so?[5]

Wittgenstein focuses on edge cases like these, cases where we are no longer sure that everybody agrees about what counts as what, in order to help us stop treating this mass agreement as *given*. In a shared culture, there is indeed mass agreement in what counts as what. But this mass agreement is an *achievement*, something painfully accomplished by constant communication and teaching, a fragile accomplishment that is always in need of renewal. For Wittgenstein, as Cavell reads him, mass agreement in counting-as activity is not something that should be presupposed at the beginning of philosophical activity, but is instead rather *something to be explained*.

> I find my general intuition of Wittgenstein's view of language to be the reverse of the idea many philosophers seem compelled to argue against in him: it is felt that Wittgenstein's view makes language too public, that it cannot do justice to the control I have over what I say, to the innerness of my meaning. But my wonder, in the face of what I have recently been saying, is rather how he can arrive at the completed and unshakable edifice of shared language from within such apparently fragile and intimate moments—private moments—as our separate counts and out-calls of phenomena, which are after all hardly more than our interpretations of what occurs, and with no assurance of conventions to back them up.[6]

Instead of an abstract "x-counts-as-y" relation that suppresses the agent performing the counting-as activity and that presupposes communal agreement, Wittgenstein wishes us to start with an *individual* agent counting something as something. It is this same counting-as activity that is needed, I claim, to understand Kant's project in the First Critique.

2.2 FROM DERIVATIVE TO ORIGINAL INTENTIONALITY

Consider the humble barometer, a simple sensory device that can detect changes in atmospheric pressure. If the mercury rises, this means the atmospheric pressure is increasing; if the mercury goes down, the pressure is

decreasing. Now *we count* the mercury's rising as the machine responding to the atmospheric pressure. We count, in other words, a process that is internal to the instrument (the mercury rising) as representing changing properties of an external world (atmospheric pressure increasing). But although *we* count the internal process as representing an external process, the *barometer itself* does not. The barometer is incapable of counting the internal process as a representing because—of course—it is incapable of counting anything as anything. The barometer does not, in other words, have original intentionality. *We* might interpret some of its activities as representations, but *it* does not.

The distinction between **original** and **derivative intentionality** comes from Haugeland.[7] Intentionality is derivative if something has it because it is conferred on it by something else (by the agent who is doing the counting-as):

> At least some outward symbols (for instance, a secret signal that you and I explicitly agree on) have their intentionality only derivatively—that is, by inheriting it from something else that has the same content already (e.g., the stipulation in our agreement). And, indeed, the latter might also have its content only derivatively, from something else again; but obviously, that can't go on forever. Derivative intentionality, like an image in a photocopy, must derive eventually from something that is not similarly derivative; that is, at least some intentionality must be original (non derivative).[8]

We can distinguish between derivative and original intentionality using the activity of counting-as:[9]

- *x* has derivative intentionality in representing *p* if an agent *y* (distinct from *x*) counts *x*'s activity as *x*'s representing *p*

- *x* has original intentionality in representing *p* if *x* himself counts *x*'s activity as *x*'s representing *p*

What distinguishes an agent with original intentionality from a mere sensory instrument is that the former *counts its own sensings* as representations of a determinate external world:

> There is no doubt whatever that all our cognition begins with experience; for how else should the cognitive faculty be awakened into exercise if not through objects that stimulate our senses and in part themselves produce representations, in part bring the activity of our understanding into motion to compare these, to connect or separate them, and thus to *work up the raw material of sensible impressions into a cognition of objects that is called experience*?[10]

Original intentionality, in other words, is a type of activity interpretation. Just as I can count his moving the horse-shaped wooden piece from one square to another as his moving his knight to king's bishop three, just so I can count the perturbations of my sensory instruments as my representing a determinate world.[11]

Searle makes a similar distinction between intrinsic and derived intentionality.[12] He claims that only a certain type of biological organism can achieve original intentionality. This paper argues, by contrast, that a *computational* agent built to satisfy a Kant-inspired cognitive architecture *is capable of achieving original intentionality*. It doesn't matter what it is made of as long as it achieves the necessary structural organization.

2.3 FROM SENSORY AGENTS TO COGNITIVE AGENTS

A **sensory agent** is some sort of animal or device, equipped with sensors, whose actions depend on the state of its sensors. It might have a temperature gauge, a camera with limited resolution, or a sonar that can detect distance. The sensory agent is continually performing what roboticists call the sense-act cycle: it detects changes to its sensors, and responds by bodily movements.

A thermostat, for example, is a simple sensory agent. When it notices that the temperature has got too low, it responds by increasing the temperature. The thermostat has a sense-act cycle, but it does not experience the world it is responding to. *We* count the perturbations of its gauge as representations of the temperature in the room it is in, but *it* does not. The gauge movements count as temperature representations *for us*, but not *for the thermostat*. Nothing counts as anything for the thermostat. It just responds blindly.

By contrast, a **cognitive agent** is a sensory agent with original intentionality, who *counts* his sensings *as* his representing an external world. He interprets his own sensory perturbations as his representation of a coherent unified world of external objects, interacting with each other. This world contains one particular distinguished object, with sensors, that the cognitive agent counts as his *body*, and he counts his sensings *as* the stimulation of his body's sensors by interaction with the other objects.

One of Kant's fundamental questions in the First Critique is:

> *What does a sensory agent have to do, in order for it to count its own sensory perturbations as experience, as a representation of an external world?*

What, in other words, must a sensory agent do to be a cognitive agent?

Note that this is a question about intentionality—not about knowledge. Kant's question is very different from the standard epistemological question:

> Given a set of beliefs, what else has to be true of him for us to count his beliefs as knowledge?

Kant's question is *pre-epistemological*: he does not assume the agent is "given" a set of beliefs. Instead, we see his beliefs as an *achievement* that cannot be taken for granted, but has to be *explained*:

> Understanding belongs to all experience and its possibility, and the first thing that it does for this is not to make the representation of the objects distinct, *but rather to make the representation of an object possible at all*.[13]

Kant asks for the conditions that must be satisfied for the agent to have any possible cognition (true *or* false).[14]

2.4 CLARIFICATIONS

Kant's question, in the first person, is

> *What must I do, in order to count these sensory perturbations as my experience?*

I wish to make two clarifying points. First, counting-as can be applied to objects or activities. We can count this wooden-shaped piece as a knight (counting an object as another object), or we can count this physical pushing movement as moving one's knight to king's bishop three (counting an activity as another activity). In what follows, I am talking primarily about counting-as applied to *activities*, not objects. It is not that we count this group of sensors as a red triangle, but rather that we count this plurality of sensor *activity* as *representing* a red triangle.

Second, the activity that is counted-as is not an individual sensor's activity, but is typically a large *plurality* of sensory perturbations. We have a huge array of independent sensors (including 6 to 7 million cone cells in each eye). When we count sensory perturbations as representing something, we are interpreting a large plurality of such sensings. Consider, by way of analogy, a beginner dance-class: under what circumstances does this flurry of bodily movements, this plurality of limb activities, count as waltzing?

2.5 THE AMBITION OF THE KANTIAN QUESTION

Some activity can be explained in terms of other activity. Getting married, for example, is not a fundamental primitive form of activity. Instead, we *count* saying "I do," under certain wedding-related circumstances, *as* getting married. We do not treat castling in chess as a fundamental primitive form of activity. Instead, we *count* moving two pieces, under certain chess-related circumstances, *as* castling. We explain activity *a* by counting activity *b*, under certain circumstances, as performing activity *a*.

An ingrained assumption of pre-critical philosophy was that the *having of thoughts* is a fundamental activity that is *not* in need of a count-as explanation. There may be (at some future time) a physical explanation of this activity in terms of neuronal firings etc.—but there is no counts-as explanation of this activity. This is what it means to say that thoughts are "given": we do not need to worry about the origin of these thoughts—we just assume that they are handed to us somehow.

Both empiricism and rationalism subscribe to a version of this pre-critical assumption. Empiricists assume that the mind is already capable of having intuitions (constructing pre-conceptual representations of objects), and tell a story explaining how the mind is able to get from intuitions to conceptual thoughts. Rationalists assume that the mind is already capable of thinking conceptual thoughts, and tell a story explaining how the mind is capable of getting from conceptual thoughts to empirical intuitions. Both strategies treat some form of thinking as primitive: as not needing a counts-as explanation.

Kant denies this assumption. He claims, to anticipate, that *all thought* can be explained by a rule-constructing agent following a procedure that satisfies certain constraints. If the agent constructs and applies rules in a certain manner, satisfying certain constraints, then his rule-following activity *counts as* his having intuitions, forming concepts, judging, and thinking about an external world.

Kant's core claim is

> *I count this plurality of sensings as my experience if I combine them together in the right way*

What, then, does Kant mean by "combine," and what does he mean by "the right way"?

In section 3, I will describe what Kant means by combining. To anticipate, there are two types of combination, achieved by applying two types of rules (rules of composition and rules of connection).

3 COMBINATION AS RULE APPLICATION

The activity at the heart of Kant's theory is the activity of **combination**, of bringing cognitions together, "running through and holding together this manifoldness."[15]

Now this activity of mental combination may seem frustratingly metaphorical or ill-defined. As Wolff notes:

> The inadequacies of such locutions as "holding together" and "connecting" are obvious, and need little comment. Perceptions do not move past the mind like parts on a conveyor belt, waiting to be picked off and fitted into a finished product. There is no workshop where a busy ego can put together the bits and snatches of sensory experience, hooking a color to a hardness, and balancing the two atop a shape.[16]

Similarly, Wittgenstein writes:

> How does one teach a child (say in arithmetic) "Now take these things together!" or "Now these go together"? Clearly "taking together" and "going together" must originally have had another meaning for him than that of seeing in this way or that.[17]

There are two types of combination activity.[18] The first is composition: combining intuitions using the part-of relation. For example: if this configuration of sensors is turned on, then I count their being-on as representing a nose. Or: if this pattern of sensors counts as representing a nose, and this other pattern counts as representing an eye, then the aggregate pattern of sensors counts as representing a face. The second type of combination activity is **connection**: subsuming intuitions under marks.[19] For example: if this is

a nose, then it cannot be an ear. Or: if this is a dog, then it must be an animal.

3.1 COMBINATION CAN ONLY BE PERFORMED INDIRECTLY VIA THE CONSTRUCTION AND APPLICATION OF RULES

But, although this combining activity is fundamental, it cannot, according to Kant, be performed directly by the agent. The agent cannot just bring representations together *willy-nilly*. Combining is not something he can *just do*. On the contrary, the *only way*, according to Kant, that the agent can perform the activity of combination is by *applying general rules that it has constructed*. This is Kant's surprising claim. We are used to thinking of social activity as constituted (e.g., a certain set of sounds counts, under the right conditions, as the request to shut the door). But we are not so used to thinking of fundamental *mental* activity as similarly constituted.

There are two types of rule corresponding to the two types of combination.[20] **Rules of composition** are rules for combining parts into wholes, producing a part-whole graph united under one element: the totality. A rule of composition produces, if it applies, a defeasible rule permitting the agent to group intuitions together.[21] For example, if you count this group of sensings as representing an ear, and this group of sensings as representing a nose, then you may count this aggregate group of sensings as representing a face. Whether or not the rule-following agent makes use of this permission will depend on his concomitant commitments.

Rules of composition are described by *defeasible* conditionals. Wittgenstein stresses the defeasibility of such conditionals when discussing what counts as a friendly face:

> When we notice the friendly expression of a face, our attention, our gaze, is drawn to a particular feature in the face, the "friendly eyes," or the "friendly mouth etc. . . . *It is true that other traits in this face could take away the friendly character of this eye*, and yet in this face it is the eye which is the outstanding friendly feature.[22]

This is defeasibility in action. In this situation, the features of this eye counts as his having a friendly face; but in another situation, the same features plus some other additional facial features might count as something entirely different: mocking cruelty, for instance.

Rules of connection produce *obligations* to group representations under a mark.[23] So, for example, if we count this structure as a nose, then we *must* also count it as a facial part—and if we count it as a nose, then we *must not* count it as an ear.

The activities of combination, then, are themselves constituted activity:

- Composing (combining intuitions together using part-of) just is applying a rule of composition
- Connecting (subsuming intuitions under marks) just is applying a rule of connection

The striking Kantian claim is that this activity of combination is not a self-sufficient action. Rather, it is like moving your knight to king's bishop three: it is something you can only do indirectly *by doing something else*—by pushing a wooden object in a certain direction. Similarly, requesting Bob to shut the door is not something you can just do: you can only do it by doing something else (perhaps by uttering a sequence of sounds, or by pointing at the door; there are an infinite number of different actions that could constitute such a request, but you have to do *one* of them—requesting is not something primitive you can do on your own).

All the agent can do is *construct* general rules of the above form, permitting or obligating him to combine representations in a certain way, and then *apply* these rules, thus *indirectly* performing combinations via the construction and application of rules.

This claim appears throughout the First Critique:

> all empirical time determination must stand under rules of general time determination.[24]

In other words, the activity of time-determination can only be performed indirectly by applying a general rule:

> everything (that can even come before us as an object) necessarily stands under rules, since, without such rules, appearances could never amount to cognition of an object.[25]

Again:

> Thus we think of a triangle as an object by being conscious of the composition of three straight lines *in accordance with a rule* according to which such an intuition can always be exhibited.[26]

In other words, the activity of seeing the three lines as a triangle is only achieved indirectly by the application of a general rule for counting certain connected triads of lines as triangles.

Why can't a cognitive agent perform the activity of combination *directly*, without needing to construct and then apply a rule? The answer is that combining without rules would not satisfy the condition of *unification* at the heart of K's theory. The unification condition is a set of constraints on the construction and application of *rules*, and so can *only* be applied to a rule-following agent. Arbitrary combination of cognitions that was unguided by rules would not produce a unity of experience that I could call *mine*. If I could combine representations into intuitions without rules, then there would be no *self* to have the intuitions.

At the beginning of the B Deduction, Kant writes:

> *all combination*, whether we are conscious of it or not, whether it is a combination of the manifold of intuition or of several concepts, and in the first

> case either of sensible or non-sensible intuition, *is an action of the understanding*.[27]

If we recall that the understanding is the capacity for constructing and applying rules,[28] then it follows that the *only* way in which we can combine representations together is via the construction and application of rules. Again:

> Thus the original and necessary consciousness of the identity of oneself is at the same time a consciousness of an equally necessary unity of the synthesis of all appearances in accordance with concepts, i.e., in accordance with rules that not only make them necessarily reproducible, but also thereby determine an object for their intuition, i.e., the concept of something in which they are necessarily connected; for the mind could not possible think of the identity of itself in the manifoldness of its representations, and indeed think this a priori, if it did not have before its eyes the identity of its action, which subjects all synthesis of apprehension (which is empirical) to a transcendental unity, and first makes possible their connection in accordance with a priori rules.[29]

In other words: the unity of the self is not something that we perceive. What we perceive are objects. If we are going to achieve unity of the self, it must be through something that persists through the sensory flux. What persists are the constraints on the rules we apply.

Suppose, to take the contrapositive, that we combine our sensings together without using rules. Suppose we just combine representations willy-nilly. Then the combined representations will just be a "mere play," "less even than a dream."[30]

The Kantian rule-following agent is continually constructing the very software that it will then execute.[31] It is always constructing rules, and then interpreting those rules. In fact, the *only* way that it can perceive *anything* is by applying rules it has already constructed in order to make sense of the incoming barrage of sensations.[32]

Instead of having a primitive ability to combine representations at will, the rule-following agent can *only* do so when it has a rule which says that it may or must do so.

The Kantian rule-follower, then, is a norm-giving agent who solemnly sets down rules that he will then obediently follow. He only allows himself to perform acts of mental combination if these acts are shown to be permitted[33] by rules he has antecedently accepted.

4 CONCERNS WITH RULE-FOLLOWING ACCOUNTS OF INTENTIONALITY

4.1 WHAT DOES KANT MEAN BY A "RULE"?

What does a sensory agent have to do to count his sensings as representations of an external world? Kant's answer is that he must construct and apply rules of combination satisfying various constraints. At the very center of Kant's theory is the notion of a rule-following agent.

But Kant's appeal to rule-following has been criticized in multiple ways. Rules have been seen as too explicit, or too rigid, to do the work that Kant requires of them. At the very least, we need a clear sense of what Kant means by "rule" if we are to make sense of his theory. What does Kant mean by "rule"?

Let us start by stating what a rule is, and then clarifying by emphasising what it is *not*. A rule, for Kant, is something general, something that applies in many different situations.[34] When it does apply, a rule results in a *norm* becoming operative: a certain activity is now permitted or obligatory. For example, if such and such circumstances hold, you *may* combine this sensor and that sensor under the mark EAR-14.

A **rule**, then, is a norm, operating under a condition:

> Now, however, the representation of a universal condition in accordance with which a certain manifold (of whatever kind) *can* be posited is called a *rule*, and if it *must* be so posited, a law.[35]

Here, Kant is clear about the *generality* in the "universal condition" and the normative status (permission/obligatory) in the "can"/"must."

Next, let us look at what a rule is *not*. Firstly, a rule is not an *explicit*, *linguistically formulated conditional*. In order to *describe* a rule, we have to use language—but that does not mean, of course, that the rule is essentially linguistically formulated. The rule is not an explicit sentence, but an implicit "procedure for generating a sensible intuition" that the sentence describes.[36] If the rule was merely an explicit, linguistically-formulated conditional, then there would be a further question of whether the linguistically formulated antecedent itself applies. If answering this further question itself required a rule, then we would have a regress of rule-application.[37]

Secondly, a rule is not a *disposition*. A disposition is some sort of probabilistic counterfactual, formulated as: if such and such conditions hold, there is a certain *probability* that a particular action will be performed. But when a *rule* applies, there is a *norm* stating that the action *must* (or *may*) be performed.

Much of Wittgenstein's *Investigations* is concerned with showing the problems with various inadequate conceptions of rules. I hope I have said enough to distinguish Kant's sense of rule from explicit conditionals and from mere regularities.[38]

One Wittgenstinian concern with rules is that they are too rigid and inflexible to capture real inferential patterns. Suppose, for instance, we want to formulate a general rule that birds fly.—Well, not all birds fly. Penguins don't fly.—But not all penguins don't fly: magic penguins do fly.—But not all magic penguins fly: magic penguins who have been trapped in a cage don't fly. There are an infinite number of exceptions for any rule. If we have to specify all the exceptions manually, our task will never be completed.[39] The proper response to this is to admit that yes, of course, a rule has an indefinite number of exceptions, but to deny that these exceptions need to be stated explicitly in the antecedent of the rule. A rule should be *defeasible*—it states that, everything else being equal, if the antecedent holds, then the consequent holds. But the conditional is not strict implication but defeasible implication. We can capture the way some rules override others by placing a partial ordering on the defeasible rules. If our rules are defeasible, and we can specify which rules override which others using a *partial ordering*, then we don't need to specify all the exceptions.

Another Wittgenstinian concern with rules is that they seem incapable of capturing penumbral concepts. Take, for example, Wittgenstein's famous example of "game." What it is to be a game is not captured once and for all by a set of definite rules. Rather, it is a family resemblance concept: some paradigms (chess, baseball) are central, while others (patience, ring-a-ring-a-roses) are peripheral. Patience is *less* of a game than baseball, but a game nonetheless. How can we capture penumbral concepts in a rule-based architecture? If we cannot capture them, this would be a serious problem for the account proposed. But it has recently been pointed out that a Bayesian mixture model over discrete rule sets *can* capture penumbral concepts rather well.[40] The idea is that the inductive learner is trying to construct a set of rules that best explains the data he has received. There are various sets of rules that capture the data with different degrees of success. The Bayesian mixture model applies the various rule-sets in proportion to their posterior probabilities. So if, say, there are just three sets of rules that explain the data, and rule-set R_1 has posterior probability 0.7 and says that the newly observed instance is a game, and rule-set R_2 has posterior probability 0.2 and says the new instance is not a game, and rule-set R_3 has posterior probability 0.1 and says the new instance is a game, then the mixture model says that the new instance is a game with probability 0.7 + 0.1 = 0.8.

4.2 WHAT DOES KANT MEAN BY A "RULE-FOLLOWING PROCESS"?

But Wittgenstein has another fundamental concern with a rule-following account of intentionality that we need to address head-on. This is the worry that a rule-following account cannot accommodate the fact that no set of rules can cover all possible cases. Wittgenstein draws our attention, again and again, to cases where our rules *give out*:

> I say "There is a chair." What if I go up to it, meaning to fetch it, and it suddenly disappears from sight?—"So it wasn't a chair, but some kind of illusion."—But in a few moments we see it again and are able to touch it and so on.—"So the chair was there after all and its disappearance was some kind of illusion."—But suppose that after a time it disappears again—or seems to disappear. What are we to say now? *Have you rules ready for such cases*—rules saying whether one may use the word "chair" to include this kind of thing? But do we miss them when we use the word "chair"; and are we to say that we do not really attach any meaning to this word, because *we are not equipped with rules for every possible application of it*?[41]

Our rules for the identification of chairs cannot anticipate every eventuality, including their continual appearance and disappearance—but this does not mean we cannot recognize chairs. Or, to take another famous example, we have rules for determining the time in different places on Earth. But now suppose someone says:

> "It was just 5 o'clock in the afternoon on the sun."[42]

Again, our rules for determining the time do not cover all applications, and sometimes just *give out*. They do not cover cases where we apply time of day on the sun. Wittgenstein's vision is that, since any set of rules is inevitably limited and partial, we must always continually improvise and update. But this vision is fully compatible with the rule-following Kantian agent, as I have described him. Such an agent is *continually constructing* a new set of rules that makes best sense of his sensory perturbations. It is not that he constructs a set of rules, once and for all, and then applies them rigidly and unthinkingly forever after. Rather the process of rule construction is a continual effort.

Kant describes an *ongoing process* of constructing and applying norms to make sense of the barrage of sensory stimuli:

> There is no unity of self-consciousness or "transcendental unity of apperception" apart from this effort, or conatus towards judgement, *ceaselessly affirmed and ceaselessly threatened with dissolution* in the "welter of appearances."[43]

Kant's rule-following agent is continually constructing such norms, so as to best make sense of the barrage of sensory stimuli. If he were to cease constructing these rules, he would cease to be a rule-following agent, and would be merely a *machine*.

In *What is Enlightenment?* Kant is emphatic that the cognitive agent must never be satisfied with a statically defined set of rules—but must always be modifying existing rules and constructing new rules. He stresses that adhering to any statically defined set of rules is a form of self-enslavement:

> Precepts and formulas, those mechanical instruments of a rational use, or rather misuse, of his natural endowments, are the ball and chain of an everlasting minority.[44]

Later, he uses the term "machine" to describe a cognitive agent who is no longer open to modifications of his rule-set.

He defines "enlightenment" as the continual willingness to be open to new and improved sets of rules. He imagines what would happen if we decided to fix on a particular set of rules, and forbid any future modifications or additions to that rule-set. He argues that this would be disastrous for society and also for the self.

In *The Metaphysics of Morals*, he stresses that the business of constructing moral rules is an ongoing never-ending task:

> Virtue is always in progress and yet always starts from the beginning. It is always in progress because, considered objectively, it is an ideal and unattainable, while yet constant approximation to it is a duty. That it always starts from the beginning has a subjective basis in human nature, which is affected by inclinations because of which *virtue can never settle down in peace and quiet with its maxims adopted once and for all* but, if it is not rising, is unavoidable sinking.[45]

Just as for moral rules, just so for cognitive rules: Kant's cognitive agent is always constructing new rules to make sense of the pattern, a pattern that is new in every moment.

This sort of rule-following model has its critics. Some of Wittgenstein's remarks, for example, are often interpreted as denying the possibility of any sort of rule-following account:

> We can easily imagine people amusing themselves in a field by playing with a ball so as to start various existing games, but playing many without finishing them and in between throwing the ball aimlessly into the air, chasing one another with the ball and bombarding one another for a joke and so on. And now someone says: The whole time they are playing a ball-game and following definite rules at every throw.[46]

Now there is a crucial scope ambiguity here. Is Wittgenstein merely denying that there is a set of rules that captures the ball-play at every moment? Or is he making a stronger claim, claiming that there is some moment during the ball-play that cannot be captured by any set of rules at all? I think the weaker claim is more plausible: we make sense of the world by applying rules, but we need to continually modify our rules as we progress through time. Wittgenstein's passage in fact continues:

> And is there not also the case where we play and make up the rules as we go along? And there is even one where we alter them, as we go along.

Here, he does not consider the possibility of there being activity that cannot be explained by rules—rather, he is keen to stress the *diachronic* nature of the rule-construction process.

We have considered various interpretations of "rule" and "rule-following" that are too rigid and inflexible to serve as the foundation for a model of intentionality.

I have claimed that, suitably interpreted, a rule-following account can survive the accusations of inflexible rigidity, as long as the rules are interpreted as:

- conditional norms, rather than explicit linguistically formulated conditionals
- conditional norms, rather than probabilistic dispositions
- defeasible conditionals, rather than strict entailments
- defeasible conditionals under a partial ordering, rather than conditionals with explicit exceptions

I have further argued that our rule-following agent must be continually expanding and modifying his rule-set: the construction of rules is an ongoing activity that we must continue forever.

5 CONSTRUCTING AND APPLYING RULES

The rule-following agent can perform just two types of basic activity. He can construct a rule, and he can apply a rule he has already constructed. I shall consider these two activities in turn.

5.1 CONSTRUCTING RULES

Kant says it is the job of the faculty of **understanding** to construct rules:

> We have above explained the understanding in various ways—through a spontaneity of cognition (in contrast to the receptivity of the sensibility), through a faculty of thinking, or a faculty of concepts, or also of judgements—which explanations, if one looks at them properly, come down to the same thing. Now we can characterise it as the *faculty of rules*. This designation is more fruitful and comes closer to its essence. Sensibility gives us forms (of intuition), but the understanding gives us rules. It is always busy poring through the appearances with the aim of finding some sort of rule in them. . . . The understanding is thus not merely a faculty for making rules through the comparison of the appearances; it is itself the legislation for nature.[47]

Recall that there are two types of rule (rules of composition, and rules of connection), so there are two types of rule construction. Constructing rules of composition is forming perceptual rules, rules of apprehension for counting particular configurations as parts of objects. For example, the agent adds a new rule that, if some of its sensors satisfy such and such a condition, it may count them as representing an ear.

Constructing rules of connection is forming concepts or making judgments. Forming a concept is constructing a set of rules that describe the inferential connections between this concept and others. So, for example, to form the concept of "tree," we need rules of composition for saying under what sensory conditions we can count an intuition as a tree. But we also need rules of connection for linking this

concept with others. So if we count it as a tree, we must also count it as a plant, and must not count it as a biscuit. Some of the connection rules involved in characterising a concept do more than simply state that one concept is a sub-concept of another, or that one concept excludes another. Some of them relate the concept to another concept only conditionally—dependent on the existence of external factors. For example: "If the weather gets cold, trees lose their leaves," "If a tree gets no water, it perishes."[48] Some of the conceptual inference rules, in Kantian terms, are hypothetical rather than categorical.

Constructing rules of connection is also what is involved in making judgments. If we form the judgment that "All men are mortal," this is just to adopt the rule of connection: if I count a cognition as a man, then I must also count it as mortal. But this inferential understanding of judgment applies to categorical statements just as much as to hypothetical ones: to form the judgment that "Caesar is a general" just is to adopt the rule: if I count a cognition as Caesar, then I must also count it as a general.

This is why Kant says that the faculty of constructing rules is also the faculty of concept-formation and judging: both concept-formation and judging are just special cases of the more general ability to construct rules.[49]

5.2 APPLYING RULES

Next, I shall turn to the process of *applying* the rules that the understanding has constructed. Kantian rules, as defined above, are norms operating under a condition. Rules are not explicit linguistic conditionals, where there is a further question whether the antecedent applies. Rather, the rule is itself responsible for determining when it applies. It contains a procedure for determining whether or not it applies. The rule-as-procedure applies itself. If the rule applies in a particular situation, a norm is operative: either the agent must combine the representations under a certain mark (if the rule is a rule of connection), or it may do so (if it is a rule of composition). If it is a rule of composition, then all the agent knows is that he *may* perform the combination activity—he does not have to do so. Consider, for example, Jastrow's famous duck-rabbit (Figure 1). Focus on the lines on the left of the image. We have two rules of composition that apply to these lines: we can count these lines as a mouth, or as a pair of ears. Now there is a rule of connection that prevents us from applying both: if something is a mouth, then it is not a pair of ears. We may apply either rule of composition—but we must not apply both. What makes us decide which to apply?

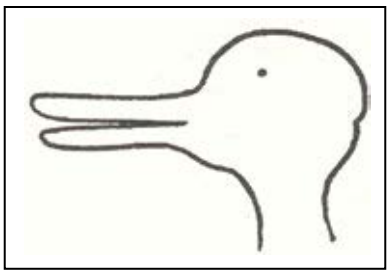

Figure 1. Jastrow's famous duck-rabbit.

Kant argues convincingly that it cannot be a further *rule* that tells us which to apply:

> Now if it [general logic] wanted to show generally how one ought to subsume under these rules, i.e., distinguish whether something stands under them or not, this could not happen except once again through a rule. But just because this is a rule, it would demand another instruction for the power of judgement, and so it becomes clear that although the understanding is certainly capable of being instructed and equipped through rules, the power of judgement is a special talent that cannot be taught but only practiced. Thus this is also what is specific to so-called mother-wit, the lack of which cannot be made good by any school.[50]

If we needed rules to determine which rules to apply, then those determining rules would themselves need further rules to determine their application, and so on, generating a vicious regress.

Kant defines the **imagination** as the faculty responsible for applying the rules that the understanding has constructed. As the duck-rabbit picture shows, the imagination has some *choice* about how to apply the rules of composition:

> Now since all of our intuition is sensible, the imagination, on account of the subjective condition under which alone it can give a corresponding intuition to the concepts of understanding, belongs to *sensibility*; but insofar as its synthesis is still an exercise of spontaneity, which is determining, and not, like sense, merely determinable, and can thus determine the form of sense a priori in accordance with the unity of apperception, the imagination is to this extent a faculty for determining the sensibility a priori.[51]

The imagination belongs to sensibility because the only things it can operate on are the sensings that sensibility has provided. But it belongs to spontaneity in that it has a choice about which rules of composition to apply.

This is why Kant says that both understanding and imagination involve *spontaneity*—the understanding has a choice about which rules to construct; and then, once it has constructed them, the imagination has a further choice about which rules of composition to apply:

> It is one and the same spontaneity that, there under the name of imagination and here under the name of understanding, brings combination into the manifold of intuition.[52]

Note that it is only when applying rules of *composition* that the imagination has choice about which to apply. When it comes to applying rules of *connection*, the rule-following agent is *obligated* to perform the required mental activity.

5.3 HAVING AN INTUITION VERSUS FORMING A JUDGMENT

The rule-following agent has two distinct types of thought:

- Thinking an object (having an intuition)
- Thinking a proposition (forming a judgment)

These two distinct types of representation are achieved by two distinct types of activity:

- An intuition is the representation formed by the activity of combination, by *applying* rules it has already constructed
- A judgment is the representation formed by *constructing* a rule

Sellars clarifies the distinction between the two types of representation by using different types of syntactic units: an intuition (thinking an object) is represented by a noun-phrase, while a judgment (thinking a proposition) is represented by a sentence. He starts by noting that some philosophers interpret "seeing as" in terms of a conjunction of a perception and a thought:

> This suggested to some philosophers that to see a visual object as a brick with a red and rectangular facing surface consists in seeing the brick and believing it to be a brick with a red and rectangular facing surface:
>
> > *This is a brick which has a red and rectangular facing surface*
>
> Notice that the subject term of the judgement was exhibited above as a bare demonstrative, a sheer this, and that what the object is seen as was placed in an explicitly predicate position, thus "is a brick which has a red and rectangular facing surface."[53]

He prefers instead to characterize intuitions by noun-phrases:

> I submit, on the contrary, that correctly represented, a perceptual belief has the quite different form:
>
> > *This brick with a red and rectangular facing surface*
>
> Notice that this is not a sentence but a complex demonstrative phrase. In other words, I suggest that in such a perceptually grounded judgement as:
>
> > *This brick with a red and rectangular facing surface is too large for the job at hand*
>
> the perceptual belief proper is that tokening of a complex Mentalese demonstrative phrase which is the grammatical subject of the judgement as a whole. This can be rephrased as a distinction between a perceptual taking and what is believed about what is taken.[54]

In my terms, there are two activities:

- combining/apprehending/intuiting (i.e., applying rules)
- forming judgments (i.e., constructing rules)

Combining produces an intuition (thinking an object) which is described by a *noun-phrase*, such as

> *This brick with a red and rectangular facing surface*

Constructing a rule produces a *judgment* (thinking a proposition) which is described by a *sentence*, such as

> *This brick with a red and rectangular facing surface is too large for the job at hand*

The underlying rule that is adopted for a judgment such as this could be expressed as something like

> For any intuition, if you count it as the same brick as this brick with a red and rectangular facing surface then you must also count it as too large for the job at hand

Kant believed that all judgment-formation is rule-adoption:

> Judgements, when considered merely as the condition of the unification of given representations in a consciousness, are rules.[55]

This claim is easiest to see in the case of universally quantified judgments. Judging that "All metals are divisible" just is adopting the conditional norm

> If you count an intuition as a metal, then you must also count it as divisible.

But Kant did not just analyse universally quantified judgments in terms of rule-adoption—he applied this account of judging as rule-formation consistently across the board to *all types of judgment*. Forming the singular judgment that "Caius is mortal" just is adopting the rule

> If you count an intuition as Caius, then you must also count it as mortal

This applies equally to sentences involving demonstratives: forming the judgment that "This brick with a red and rectangular facing surface is too large for the job at hand" just is adopting the rule

> For any intuition, if you count it as the same object as this brick with a red and rectangular facing surface then you must also count it as too large for the job at hand

If, later, I have another intuition of the brick (perhaps from another angle, and further away), and count this intuition as

representing the *same object* as my earlier intuition, then I must apply the rule I have adopted, and also count this new intuition as representing a brick that is too large for the job at hand. This new combination, which I am forced to perform because of the rule I have adopted, is a new intuition, that would be described by a noun-phrase rather than a sentence:

> *This brick, with a red and rectangular facing surface, which is too large for the job at hand*

5.4 CONSTRAINTS ON THE CONSTRUCTION AND APPLICATION OF RULES

Kant's cognitive agent is a rule induction system that makes sense of its sensory perturbations by constructing and applying rules. Given that there are two types of activity (constructing and applying rules), and two types of rule (rules of composition and rules of connection), we have a square of operations:

	Rules of Composition	**Rules of Connection**
Constructing	Forming perceptual rules	Forming concepts and judgments
Applying	Forming intuitions	Inferring further properties of objects

Recall our original questions:

> *What must I do, in order to count these sensory perturbations as my experience?*

The rule-following agent is a central part of Kant's answer:

- A sensory agent is a cognitive agent if he counts his sensings as representing an external world
- He counts these sensings as representing an external world if he combines those sensings together in the right way
- He combines his sensings together in the right way if he constructs and applies a set of rules that satisfy a set of (as yet unspecified) *constraints*

The next question, then, is *what set of constraints on the construction and application of rules are severally necessary and jointly sufficient for counting this plurality of sensory perturbations as representing an external world (and thereby achieving original intentionality)?*

In the *Schematism* and the *Principles*, Kant provides a list of constraints on the self-legislating agent, and argues that these are all and only the constraints that need to be satisfied for the agent to achieve original intentionality. These arguments are difficult and dense. I attempt to summarize them in *A Kantian Cognitive Architecture*. The basic idea is that an agent can only achieve intentionality if it combines its cognitions together into a *unity*. The only relation in which all cognitions can be unified is *time*. Unifying our cognitions in time involves four aspects: constructing moments in time, generating intermediate moments of time, providing a total ordering on moments of time, and generating the totality of time (by excluding moments that are impossible). Each of these four aspects of time determines constraints on the construction and application of rules. We move from one top-level constraint (Kant calls it the "supreme principle"), that our cognitions be unified, into four sub-constraints (the four aspects of time-determination), and from each of these four sub-constraints, we generate specific constraints on the types of rules that can be *constructed* and further constraints on the results of *applying* these rules.[56]

6 CONCLUSION

Some of the most exciting and ambitious work in recent philosophy[57] attempts to re-articulate Kantian (and post-Kantian) philosophy in the language of analytic philosophy. Now this re-articulation is not merely window-dressing.—It is not merely dressing up old ideas in the latest fashionable terminology.—Rather, analytic philosophy, when done well, achieves a new level of perspicuity. Saying it again, at this level of clarity and precision, is worth saying again.

My aim is to re-articulate Kant's theory at a further level of precision, by reinterpreting it as a specification of a *computational architecture*.

Why descend to this particular level of description? What could possibly be gained? The computational level of description is the ultimate level of precise description. There is no more precise you can be: even a mere *computer* can understand a computer program. Computers force us to clarify our thoughts. They admit no waffling or vagueness. Hand-waving is greeted with a compilation error, and a promissory note is returned, unread.

The advantage of re-articulating Kant's vision in computational terms is that it gives us a new level of specificity. This is, in fact, the final level of specificity. There is no more precise we can be.

The danger is that, in an effort to shoe-horn Kant's theory into a particular implementable system, we distort his original ideas to the point where they are no longer recognisable. Whether this is indeed the unfortunate consequence, the gentle reader must decide.

I have formalized (a particular interpretation of the first half of) the First Critique as a specification of a computational architecture. I have implemented this architecture as a computer program and tested it in two domains.

In one domain, the sensory agent has to make sense of its sensory readings in a simple two-dimensional grid world. The rule-induction agent is forced to construct rules that make sense of the barrage of sensory data. In doing so, it creates a unified cognition, combining momentary apprehensions into persisting objects that change over time, objects that change according to intelligible rules, and interact with other objects according to intelligible rules.

The second experimental domain is a verbal reasoning task. The Kantian machine is given a sequence of symbols and has to predict the next element in the series. For example: find the next element in the sequence:

a, k, b, k, k, c, k, k, k, ...

The Kantian machine makes sense of these sequences by constructing rules that, when applied, satisfy the various Kantian constraints. Surprisingly, the Kantian machine is able to achieve human-level performance in these verbal reasoning tasks.[58]

ACKNOWLEDGEMENTS

I am grateful to Barnaby Evans, Joel Smith, Marek Sergot, Murray Shanahan, Peter Boltuc, and Tom Smith for helpful discussions.

NOTES

1. This paper is a companion to Richard Evans, *A Kantian Cognitive Architecture* (IACAP, 2016), which goes into the technical details.
2. Wittgenstein,*Investigations*, §206.
3. Ibid., §185.
4. For Wittgenstein, the community of "we" *just is* the set of individuals who count-as in the same way.
5. Wittgenstein,*Investigations*, §200. My emphasis.
6. Stanley, Cavell, *The Claim of Reason* (Oxford: Oxford University Press, 1979), 36.
7. John Haugeland, "The Intentionality All-stars," *Philosophical Perspectives* (1990): 383–427.
8. Ibid., 385.
9. Note that I am not *defining* intentionality in terms of the activity of counting-as (which would be uninformative). Rather, I am using counting-as to *distinguish* between original and derivative intentionality. Later, counting-as will itself be explicated in terms of the construction and application of rules.
10. Immanuel Kant, *The Critique of Pure Reason*, B Edition, 1. My emphasis.
11. In other words, we can only represent a world because we can count some activity as mental activity. Therefore, the ability to count activity as intentional (representational) behavior is necessary to be able to think a world at all. This has interesting consequences for scepticism about others' minds. The sceptic suggests it is possible for us to be able to make sense of a purely physical world of physical activity, and asks with what right we assume that some of this activity is mental activity. But if the above is right, the capacity to count activity as mental activity is *necessary to think anything at all*—there is no intentionality-free representation of the world, in terms of bare particulars. There is always already the ability to see activity as intentional activity before we can see anything.
12. John Searle, *The Rediscovery of the Mind* (Cambridge, MA: The MIT Press, 1992).
13. Kant, *The Critique of Pure Reason*, A Edition, 199; B Edition, 244–45.
14. Ibid., A158, B197.
15. Ibid., A99.
16. Robert P. Wolff, *Kant's Theory of Mental Activity: A Commentary on the Transcendental Analytic of the Critique of Pure Reason* (Harvard University Press, 1964), 126.
17. Wittgenstein,*Investigations*, §208e.
18. See Kant, *The Critique of Pure Reason*, B201n.
19. Kant says little about what a "mark" is, given its load-bearing role in his theory. "Merkmal" is typically translated as "mark," but it can also be translated as "feature." A **mark** is not a shared linguistic symbol. It is rather what computer scientists call a "gen-sym": a generated symbol, an atomic identifier. A mark is an uninterpreted symbol, on which the only primitive operation that you are given is a procedure for testing identity: the agent can tell, when given two marks m_1 and m_2, whether or not $m_1 = m_2$. A mark, on its own, is just an uninterpreted symbol. But by constructing inferential rules that relate this mark to others, we can elevate it into a concept.
20. Kant, *The Critique of Pure Reason*, B201n.
21. See ibid.: "the synthesis of a manifold of what does *not necessarily* belong to each other."
22. *Brown Book*, 145–46.
23. See Kant, *The Critique of Pure Reason*, B201n: "the synthesis of a manifold of what does *not necessarily* belong to each other."
24. Ibid., A177, B220.
25. Ibid., B198, A159.
26. Ibid., A105.
27. Ibid., B130.
28. Ibid., A126.
29. Ibid., A108.
30. Ibid., A112.
31. In computational terms, think of a meta-interpreter that is able to construct pieces of code as *data*, and then execute these new pieces of code.
32. Kant makes the same point in the Metaphysical Deduction: "The same function that gives unity to the different representations *in a judgement* also gives unity to the mere synthesis of different representations *in an intuition*, which, expressed generally, is called the pure concept of the understanding. The same understanding, therefore, *and indeed by means of the very same actions* through which it brings the logical form of a judgement into concepts by means of the analytical unity, also brings a transcendental content into its representations by means of the synthetic unity of the manifold" [A79, B104-5]. In other words, there is only *one* process (a process of constructing and applying rules) which explains *both* how we form judgments *and* how we form intuitions.
33. An activity is shown to be permitted if there is a rule that applies that shows that you may *or must* do it. Must implies may.
34. Kant, *The Critique of Pure Reason*, A106.
35. Ibid., A113.
36. Beatrice Longuenesse, *Kant and the Capacity to Judge* (Princeton University Press, 1998), 50.
37. See, e.g., Robert B. Brandom, *Making It Explicit: Reasoning, Representing, and Discursive Commitment* (Harvard University Press, 1994), 20 ff.
38. Thereby steering between the Scylla of regulism (rules as linguistically explicit conditionals) and the Charybdis of regularism (rules as mere statistical regularities). See Brandom's *Making It Explicit*, Chapter 1, Section 3.
39. In AI, this is called the qualification problem.
40. Joshua B. Tenenbaum, *Rules and Similarity in Concept Learning* (NIPS, 1999).
41. Wittgenstein, *Investigations*, §80.
42. Ibid., §351.
43. Longuenesse, *Kant and the Capacity to Judge*, 394.
44. Immanuel Kant, "What Is Enlightenment?" *On History* (1784): 3–10.
45. Kant, *The Metaphysics of Morals* (1797) 6:409. My emphasis.
46. Wittgenstein, *Investigations*, 83.
47. Kant, *The Critique of Pure Reason*, A126.
48. Longuenesse, *Kant and the Capacity to Judge*, 145.

49. Kant, *The Critique of Pure Reason*, A126, quoted above.
50. Ibid., A133, B172.
51. Ibid., B151.
52. Ibid., B162n.
53. Wilfrid Sellars, "The Role of Imagination in Kant's Theory of Experiencei" in *Categories: A Colloquium*, ed. H. W. Johnstone, Jr. (Pennsylvania State University, 1978), 455.
54. Ibid., 456.
55. Kant, *Prologomena to Any Future Metaphysics*, 1783, §23.
56. For details of the precise constraints involved, see Evans, *A Kantian Cognitive Architecture*; Longuenesse, *Kant and the Capacity to Judge*; and Wayne Waxman, *Kant's Anatomy of the Intelligent Mind* (Oxford University Press, 2013).
57. Robert B. Brandom, *Between Saying and Doing: Towards an Analytic Pragmatism* (Oxford University Press, 2008); Robert B. Brandom, *From Empiricism to Expressivism* (Harvard University Press, 2015); Wilfrid Sellars, *Science and Metaphysics: Variations on Kantian Themes* (New York: Humanities Press, 1968); Sellars, "The Role of Imagination in Kant's Theory of Experience"; and Tenenbaum, *Rules and Similarity in Concept Learning*.
58. For more details of the approach and the experiments, please see Evans, *A Kantian Cognitive Architecture*.

REFERENCES

Brandom, Robert B. *Making It Explicit: Reasoning, Representing, and Discursive Commitment*. Harvard University Press, 1994.

———. *Between Saying and Doing: Towards an Analytic Pragmatism*. Oxford University Press, 2008.

Brandom, Robert B. *From Empiricism to Expressivism*. Harvard University Press, 2015.

Cavell, Stanley. *The Claim of Reason*. Oxford: Oxford University Press, 1979.

Evans, Richard. *A Kantian Cognitive Architecture*. IACAP, 2016. To appear in *Philosophical Studies*, 2017.

Haugeland, John. "The Intentionality All-stars." *Philosophical Perspectives* (1990): 383–427.

Heidegger, Martin. *Kant and the Problem of Metaphysics*. Indiana University Press, 1997.

Kant, Immanuel, and Arnulf Zweig. *Correspondence*. Cambridge University Press, 1999.

Kant, Immanuel, and Paul Guyer. *The Critique of Pure Reason*. Cambridge University Press, 1781 and 1787.

Kant, Immanuel. "What Is Enlightenment?" *On History*, 1784: 3–10.

Longuenesse, Beatrice. *Kant and the Capacity to Judge*. Princeton University Press, 1998.

———. *Kant on the Human Standpoint*. Cambridge University Press, 2005.

Mulhall, Stephen. *Inheritance and Originality: Wittgenstein, Heidegger, Kierkegaard*. Oxford University Press, 2001.

———. *On Being in the World: Wittgenstein and Heidegger on Seeing Aspects*. Routledge, 2014.

Parsons, Charles. "The Transcendental Aesthetic." *The Cambridge Companion to Kant* 3 (1992): 62.

Searle, John. *The Rediscovery of the Mind*. Cambridge, MA: The MIT Press, 1992.

Sellars, Wilfrid. *Science and Metaphysics: Variations on Kantian Themes*. New York: Humanities Press, 1968.

———. "Some Remarks on Kant's Theory of Experience." *The Journal of Philosophy* (1967): 633–47.

———. "The Role of Imagination in Kant's Theory of Experience." In *Categories: A Colloquium*, edited by H. W. Johnstone, Jr. Pennsylvania State University, 1978.

Tenenbaum, Joshua B. *Rules and Similarity in Concept Learning*. NIPS. 1999.

Waxman, Wayne. *Kant's Anatomy of the Intelligent Mind*. Oxford University Press, 2013.

Wittgenstein, Ludwig, and Peter Docherty. *The Blue and Brown Books: Preliminary Studies for the "Philosophical Investigations."* Wiley-Blackwell, 1991.

Wittgenstein, Ludwig. *Philosophical investigations*. John Wiley and Sons, 2010.

Wolff, Robert P. *Kant's Theory of Mental Activity: A Commentary on the Transcendental Analytic of the Critique of Pure Reason*. Harvard University Press, 1964.

I Am, Therefore I Think

Don Perlis

UNIVERSITY OF MARYLAND

ABSTRACT

I argue that reflexive self-knowledge is the basis of all knowledge, and that a *reflexive-self* formulation of mind and consciousness—perhaps unlike formulations couched in more vague in terms of *subjectivity*, *felt experience*, or *what it's like to be*—appears that it might be studied fairly directly as a kind of engineering problem.

INTRODUCTION

Much has been written on the nature of knowledge, and its relation to mind. And the same term—*knowledge*—is used with abandon in artificial intelligence. In the Fall of 2015 I gave a series of three talks in Birmingham and Lyon, on a variety of topics including mind, meaning, and the history of AI. In subsequent discussions with Peter Boltuc, I came to think that knowledge is a powerful unifying theme to all three talks, and Peter suggested that I might turn those talks into a paper with such a focus.

As a result this paper may take what might seem like a curiously rambling back-and-forth tour covering many areas, which I hope the reader will forgive. In addition, the paper is deliberately of a rather impressionistic style, not a formal analytic argument. My view is that the ideas and concepts are in many cases vague enough (and likely will remain so until such time as we actually scientifically *solve* much of the mind-body problem) that the most fruitful approach is to suggest promising avenues for research rather than precise definitions and sharp derivations. So hopefully my paper can be read as providing suggestive hints; and that is why I employ a rather loose style that jams together what traditionally are taken as distinct topics. But I think the evidence points to a promising synthesis and unifying direction for investigation across the cognitive sciences. My conclusions will be that while knowledge is an especially narrow kind of thing (essentially a form of self-knowledge) it is also very specially at the center of cognition; and that this can be studied computationally.

PART I: AI RETURNS TO ITS ROOTS; BUT WITH A GLARING CHASM

Artificial intelligence currently is a vast field largely fueled by its eye-catching applications, from the chess-playing

prowess of Deep Blue to the growing presence of robots in many walks of life.

But I think AI may now—with some very major advances under its belt—be returning to its roots as the *science of cognitive agents*, with the exalted aim of a computational understanding of the mind. Here is a brief history:

1960–1985: Youthful ambitions, during which were introduced the situation calculus, the frame problem, SHAKEY the robot, and nonmonotonic reasoning, among other advances; but also during which there was a keen interest in building agents with fairly general cognitive abilities (but which proved vastly harder than anticipated).

1985–2010: Age of specialization, in which AI splintered into many separate areas with their own conferences and journals, major applications appeared, and little was heard about cognition. Still, there was also beginning work on time-sensitive reasoning and metacognition.

2010–present: Renewed interest in cognitive agents; natural-language processing (NLP) and robotics and vision and reasoning and planning and learning starting to come together.

But something key still seems missing. Without further ado, I give my view: we need to get a computational grasp on *reflexive self*. In fact, most AI systems have no notion of self at all—no metacognition, for instance. They simply perform, and do not take themselves into account. Thus, most NLP dialog programs are not able to answer questions about what they just said, or what their words mean; they do not model themselves as agents with purposes. Yet some research does incorporate such constructs, and there does not seem to be any fundamental puzzle about this effort; metareasoning is the principal method.

Reflexive self, however, is something even more: it is the immediate present "I," the self-knowing self. One doesn't experience pain and then come to know one is in pain. The having of the pain is the knowing it. This is controversial, but not without supporters. It is to be distinguished from reflective or introspective or biographical self. We will return to reflexive self later; but notice that the notion of knowledge has crept in.

And in fact one construct central to most of AI—the so-called knowledge base (KB)—leaves much to be desired. Most AI systems have no way to relate items in their KB to what those items supposedly stand for in the world; any such relation is in the minds of the human designers. Thus in some strong sense, AI systems today do not know anything.

So this leads us to ask what knowledge is, and how it might be seen as a kind of computational process. We close this section with an aside on the holy grail of cognitive science: what is it to be conscious? From the Latin, *conscious (of)* = *with knowledge (of)*. I suggest that a (reflexive) conscious state of a subject S is a state that S knows itself to be in, where that very knowing is part of that same state. (Here state is to be regarded as a kind of process, rather than an instantaneous instant.) A not uncommon view is that self-knowledge is a more complex form of knowledge that is preceded by knowledge of things more generally. But, as will unfold later, I think this is exactly backwards.

PART II: WHAT DOES IT TAKE FOR A PIECE OF DATA TO BE KNOWN?

A knowledge base is a repository. Without something additional, there is no more reason to regard a KB item as something known (to be true) than as something false or as a mere syntactic entity without meaning. What is missing, it would seem, is a *knower* that relates the symbolic item to its meaning in the world. (This of course is the famous issue of intentionality, the directedness of cognitive items.) Boxology—putting items in boxes and labeling them as knowledge, goals, intentions, and so on—does not settle or explain anything; yet sadly it is the current standard in AI.

So, what constitutes a knower, and how can anything be known? Can one know anything? It seems easy (e.g., Hume) to doubt anything—yet Descartes insists that there is one thing one can know for sure: that one is carrying out an act of thinking. (To my mind, his further claim that therefore he exists pales by comparison.) One cannot think and not know it.

As an aside, here is a related kind of argument due to Kripke (in Naming and Necessity, pp 153-4): pain requires a subject, an entity that can feel, can *know* the pain. But mere C-fiber firings cannot supply such a feeler/knower. Thus pain (and by extension, consciousness) is more than firing of C-fibers. But (*pace* Kripke), a whole brain-full of fibers firing in the right ways might be able to supply a feeling subject. That after all is the whole question. The reason the C-fibers argument works is that no one imagines C-fibers in themselves—sending signals in one direction—is enough. Kripke's argument—if applied to the whole brain—is close in spirit to zombie-arguments, which we return to at the end.

But back to knowing, especially since consciousness seems to be a kind of knowing.

PART III: PLATO TO GETTIER AND BACK TO DESCARTES

The nature of knowledge has been discussed and argued about for millennia. The JTB—justified true belief—theory is attributed to Plato and was held in wide acceptance until Gettier's bombshell counterexamples in the 20th century. By now others have extended this almost to a cottage industry for creating such cases. The underlying issue is this: knowledge seems to require some kind of actual connection (or acquaintance, in Russell's terms) with the facts; it cannot be a lucky guess to count as knowledge. And this is supposed to be provided by the justification. But standard sorts of justifications seem to fail, surprisingly.

The best example I know of is due to Mason Remaley (who at the time was an undergraduate in my Intro to AI class). It goes like this: You have parked your car outside your office building. While you are at work in the office, unknown to

you a landslide occurs in the parking lot and carries away many of the cars, but not yours. You still believe your car is parked there, and you have excellent reason to believe this, and it is true. Yet we would be loathe to say you *know* your car is parked there.

Such Gettier-style examples have led some (e.g., Dretske) to suggest that perhaps there is no such thing as knowledge, only beliefs of varying degrees of plausibility. This may seem convincing: isn't all so-called knowledge inferred, concluded on reflection, and thus error-prone?

We return to Descartes.

Some have argued that his premise "I think"—or cognize—is flawed: How does he conclude *he* thinks? Isn't he instead merely justified in concluding that "there is thinking"? But how can he conclude this unless he somehow knows about that thinking; and how can he know about it other than by the very doing of the thinking? Consider what it would mean for there to be thinking going on, but no one at all knows about it; this would not seem to be thinking. (Indeed, one might go further and claim that the present thinking act *is* the I/self of that moment.[1]) And this does not seem to be introspective knowing—the thinking directly and immediately involves the act of knowing about itself going on. And a form of reflexive self has returned.

PART IV: PERRY AND "I"

John Perry gives a famous example of coming to realize it is he himself whose shopping cart has a torn bag of sugar making a mess. (He is wheeling his shopping-cart along the aisles of a store, going faster and faster in an attempt to notify an unknown shopper ahead of him—following the trail of sugar that is getting thicker and thicker—until he red-facedly understands.) This experience leads him to ask what knowledge he has gained in coming to this realization. And he finds it very difficult to explain the "I"—the self-known self—in terms not involving that same construct. It seems irreducible.

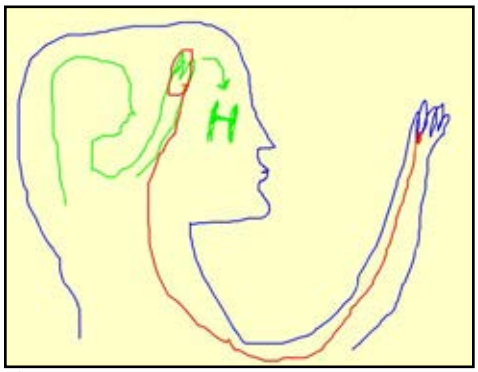

Figure 1.

Are we then stuck? Is this perhaps much the same as the explanatory gap between consciousness and physical causation? In a later section I propose a way out.

PART V: BELIEF AND MEANING

Again from Descartes, whatever knowledge is, it is self-knowledge that *precedes* other forms. And it is a strong form of knowledge, not mediated by reflection, inference, perception.

So self-knowledge precedes JTB; one first knows (oneself) and then infers regarding justifications and so on. But then what is belief? This brings us right back to boxology: a belief-box settles nothing. There has to be meaning for there to be belief. One cannot believe mere symbols; a belief is always about something.

This recalls the problem of intentionality, and also that of external reference. But Putnam's Theorem shows that there is no unique truth-preserving mapping between words/sentences and the world.[2] For any such mapping, many others do just as well. This seems to rule out any sensible account of meaning (despite Putnam's own earlier work, see below). But this argument leaves out a key component: meaning is given by *meaners*—individuals who ascribe meaning to their words and sentences. We are connected to the world via our bodies; and (at least some of) our words and thoughts are canonically connected to our bodies via specific neuronal pathways.

Indeed earlier on, both Putnam and Kripke—in the so-called causal-history theory of reference—refer to dubbers, namers, baptisers, who assign word-meanings that are then later borrowed by other members of a linguistic community.[3] They do not provide a detailed account of such initial dubbing/naming activities; but such an account would seem to be far more fundamental to language and meaning than the borrowings that flow from them. I suggest that the same bodily connection referred to above may be the basis for naming and meaning overall.

Now this is what Searle said cannot be done (by a computational system). Mere symbol processing (which is syntactic) cannot provide semantics. Unless it can. But how can it? What is being suggested here is that a self (embedded in the world) is what is needed, to supply the canonical connections.

In Figure 1, both H and the green hand (encircled in red) are internal representations, and the agent takes H to stand for that hand, which in turn is mapped neuronally and uniquely to the actual hand. So Searle is right in a way: the standard sort of symbol processing cannot supply meaning; it takes a special sort, that has self-representations.

What has this to do with knowledge? Everything! Knowledge requires meaning, hence self. And what is self? I am suggesting it is special kind of processing (that perhaps it is even like something for it to be underway): it knows itself immediately, as part of that very processing. It is the most basic form (and perhaps definition) of knowing—the self. What the self (always) knows is itself as that very process of knowing.

PART VI: MYSTERIES, OR ENGINEERING?

Have we just replaced one mystery (or two: knowledge and consciousness) with another? I think there is some progress here. For the reflexive-self formulation—perhaps unlike formulations in terms of *subjectivity, felt experience*, or *what it's like to be*—appears that it might be studied fairly directly as a kind of engineering problem.[4] By analogy, consider James Watt's *governor* that self-regulates (see Figure 2).

As increased steam-flow causes the arms to rotate with increased speed, they are pushed upwards by the vector sum of a force pulling along the arms (upward and inward) and an outward effective centrifugal force; and the new higher position then results in a steam-opening being partially closed thereby reducing the flow of steam and thus slowing the rotation, leading to an up-down cycle until a steady state is achieved. This seems magical, sense-defying, until one studies it. It does indeed have a kind of self-control. To be sure, it is not reflexive (or reflective) in any relevant sense. But it provides a compelling metaphor and food for thought. It does have dual causality—the steam causes the rotation, which also reduces the steam.

Figure 2. Watt governor.

What might be a similarly practical example of reflexivity? How about the utterance, "I am now speaking English"? Note the difference from "This sentence is in English." The latter does not refer to a speaker; there is no performance relevant to its meaning. But the former is uttered by an agent *as* it reasons about its very same uttering-in-progress. It might be tricky to implement this in a computer or robot; but it is something one can work on.

Here is another example: how can one tell one is speaking? It surely is not by hearing ones own voice. We might hear our voice and take it to be a recording. We know we are speaking by the mere fact of engaging in the action (e.g., issuing the commands to our vocal cords). The actual speech might be impeded (say by an overly dry throat), but we still know we are engaged in the process, because we are (voluntarily) undertaking it.

This too can be implemented in a robot—in fact, a robot inadvertently *taught* my research team this lesson. We had programmed it so that whenever it heard the word "Julia" it would look for her, point, and say "I see Julia." Yet often it would perform this as expected, and then go on to do it over and over every few seconds. It took us several minutes to figure out that it was responding to hearing itself say "Julia." We had not thought to provide it with a way to distinguish self-utterances from others. In fact, it had no notion of self at all. But there was ready-to-hand a solution, from neuroscience: efference copy.[5]

Efference copy is a copy of a motor command (sent to muscles) from the brain; the copy is kept in the brain, providing a kind of working memory of what the agent is doing, and which can be used to make corrections if performance deviates from goals. The most famous example is VOR—vestibular-ocular reflex—in which ones eyes rotate in their socket as ones head turns, so that one retains a stable image on the retina when gaze is fixed on an object. A similar mechanism is hypothesized to be at work in all voluntary actions. We were able to build efference copy into our robot, so that it now can distinguish between cases of hearing its own utterances and those of others; or more precisely, between cases of its undertaking or not undertaking utterances. Is this then a conscious robot, or a "self-knowing" robot in any serious sense? Surely not. But it seems headed in the right direction.

Again, if we are able to get an NLP program simply to use "I" correctly in a wide range of circumstances, we might then be a lot further along. And again, this does not seem fundamentally mysterious, rather a tough engineering problem. But this too will not be enough.

PART VII: IMAGINATION

One needs a real-time and real-space connection with the world, in order to have meaning (and thus in order to have knowledge). This is implicit in what was said above, but it needs to be brought out. Here is an example due to Patrick Winston.[6]

Consider a table-saw with the warning "Do not wear gloves when using this saw." This might be puzzling—after all, gloves are usually protective—until we visualize what might happen: in our mind's eye we see the glove being caught by a saw-tooth and then pulled (along with our hand) into the spinning blade. This inner eye is key to our ability to anticipate possibilities. Essentially it amounts to imagination, without which we would not be able to think, hence not to understand, and thus not to have knowledge (except in the unhelpful boxology sense).

This notion of thinking is not ordinary logic.[7] It combines symbolic processing with perceptual processing, perhaps akin to virtual/augmented reality and even a kind of internal self-modeling activity (e.g., seeing oneself pushing ones hand forward). Knowledge then might amount to a Cartesian awareness of self-in-mental-action. And yet, this may turn into a (self-based) engineering problem.

Perhaps then a reflexive-self based process is our only real knowledge—we can't be wrong about our own ongoing imaginative acts (Descartes; not his dualism but simply his cogito in reverse: in knowing ourselves—i.e., in self-predicating—we can be sure that we know). A so-called knowledge base is a mere storehouse of codes, not known in any useful sense until triggered into imaginative acts of anticipation.

PART VIII: SOME TENTATIVE CONSEQUENCES

A number of consequences seem to follow from the above perspective.

For one, we can suggest a conclusion about Frank Jackson's color-deprived Mary. She is supposed to know "all physical facts" about the brain, and yet never have actually been in the presence of anything red. Does she then know what it is like to see red? On the theory of knowledge being proposed here, if she knows all facts about the brain then

her brain actually triggers codes for all such facts into her imagination. And then plausibly yes, she will know the actual experience of seeing red.[8]

If this is hard to swallow, consider the following variant: Suppose you have never seen a regular pentagon, living in a world of right angles. You have read (un-illustrated) books about pentagons, describing them verbally in detail. Now will you know what a pentagon looks like? That depends on your brain's imaginative powers, which in turn depends on connections between different envisioning capacities and abilities to retain envisioned information (e.g., up to five linear elements) at once. It is a fact about pentagons and suitable brains that a certain "look" of the former can be imagined by the latter with never an external pentagon to look at; and other brains cannot. Those latter brains then do not have all the facts about brains and pentagons.

Second: In an online video, Kevin O'Regan poignantly asks (of any proposed theory that consciousness consists in X): what is it about X that makes it conscious? This is a powerful question, that leaves many theories in the dust (they offer no explanation of *felt experience*). But without intending to sound facile or glib, I suggest: self-knowingness is consciousness, and what makes it conscious is that very self-knowingness. This at least is not obviously lacking in plausibility. Self-knowingness does seem tightly linked to consciousness, and has been so linked going back at least to Descartes. Moreover, it is the sort of thing that gives us handles to work with, and might allow an engineering approach that could shine much light.

Third: Block's P-consciousness and A-consciousness now become the same thing.[9] One cannot be a knowing self and yet not know that.[10]

Fourth: In Davidson's "Knowing One's Own Mind" there is a thought-experiment in which a creature suddenly is formed in a swamp by random accident, but coincidentally molecule-for-molecule identical to an actual human (Davidson himself).[11] The Swampman will have consciousness, thoughts, feelings, etc, right off the bat in virtue of its having the identical self-engineering as a human. But (in agreement with Davidson) many of its thoughts will fail to refer externally (in the customary sense), at least until Swampman has gone on to have relevant experiences (such as meeting Davidson's friends, etc.); and some thoughts might never completely refer as long as Swampman takes himself to have Davidson's past history. But this is no great oddity; all of us have misconceptions about ourselves and the world, including failed reference.

Fifth: Determinism and freedom and time. Physics is what it is: either deterministic or not (e.g., quantum-mechanical uncertainty). But our decisions do result from a complex (physical) process that *includes* (and depends on) our deliberations. Thus we are indeed (partially) the makers of our own fate. This is not independent of physical law but rather part and parcel of it: we too are the physics, and our deliberations are no illusion; without our deliberative activity, our behavior would be far different. This may be less so in the case of minor short-term decisions such as whether to lift a finger as a clock-hand sweeps by during an experiment;[12] but for instance in deciding on whether to buy a house a great many things enter in over a long time-period, and the final moment of "choice" is a result of all the previous efforts (conscious and otherwise). We are as determined as the world (which may be determined or not), but our choices are real and effective internal parts of ourselves and of that world. And this brings us to zombies.

PART IX: ZOMBIES AND SUMMARY OF THE OUTLOOK BEING PRESENTED

We can also draw conclusions about zombies, and this will perhaps dramatize the nature of the view being presented here. To set the stage, here is a simple argument that zombies cannot exist (but I note that the literature on zombies is large and sophisticated, and it is unlikely that those who have studied this topic will be swayed by my rendering): When we say (honestly) that we are feeling a throbbing pain in our toe, it normally is the case that (this is premise #1) *we are in fact feeling such a pain*, and also that (this is premise #2) *our saying so is based causally on that felt experience itself* (that is, we would not have said so had we not been feeling the pain). But then our zombie equivalent will also say the same thing while *not* having any such experience (by definition, being a zombie). So its saying so cannot have that same causal basis.

But this is a contradiction: its neural firings are identical to ours, and since its firings are a sufficient cause for its utterance then they must also be so for ours. If we accept the two premises above, then we are forced to conclude that there can be no zombies.

Now, a zombie-loving philosopher may complain about premise #2 and say that is the issue: whether our felt experiences can cause anything physical, as opposed to being mere epiphenomena, feely-freebies, so to speak. Yet we do consult our experience—*hmm, does my toe hurt? Let me attend to how my toe feels—ah, yes—there it is, that throbbing pain, and gosh, it's getting worse*. To deny that such a statement is in part due to the existence of such a felt experience appears to deny that words have their ordinary meaning. As indeed they do not for zombies. A zombie cannot mean anything by its words, since it has no self, no "I" to take itself to intend something. And thus a zombie also cannot know anything. (This of course is a modest conclusion since we already argued that there can be no zombies.)

The above is contentious—in that it hinges on intuitions. But on the self-knowing theory discussed here, a self-knowing physical process *is* its own experience, hence again there can be no zombies. Any physical arrangement of the proposed sort already involves any associated feelings, and it will be inconceivable (once we know the details) that it could be otherwise. *Pace* Kripke (again), pain is like heat: the physics is what it is. It simply is that we do not yet have a clear enough grasp on the kind of complex interactions that can occur in a brain—any more than in 1900 chemists had a clear (or even dim) sense of self-reproducing molecules. Thus the outlook I am presenting is this: Argument is not needed; all (!) we need to do is the (hard) engineering work to discover how self-knowingness occurs.[13]

ACKNOWLEDGEMENTS

I wish to thank the following for many helpful comments in the writing of this paper (but no errors or misconceptions are to be attributed to them—I stubbornly ignored some of their advice!): Peter Boltuc (and various of his students), Justin Brody, Jean Dickason, Nick Humphrey.

NOTES

1. See D. Perlis, "Consciousness as Self-Function," *Journal of Consciousness Studies* 4, nos. 4-5 (1997): 509–25; and D. Perlis, "Five Dimensions of Reasoning in the Wild," *AAAI*, 2016.
2. H. Putnam, "Models and Reality," *Journal of Symbolic Logic* 45, no. 3 (1980): 464–82.
3. H. Putnam, "Meaning and Reference," *Journal of Philosophy* 70, no. 19 (1973): 699–711; and S. Kripke, *Naming and Necessity* (Harvard University Press, 1980).
4. Pace Chalmers (D. Chalmers, *The Conscious Mind* [Oxford University Press, 1996]), who insists consciousness has no function.
5. J. Brody, D. Perlis, and J. Shamwell, "Who's Talking—Efference Copy and a Robot's Sense of Agency," AAAI Fall Symposium, 2015.
6. P. Winston, "The Strong Story Hypothesis and the Directed Perception Hypothesis," AAAI Fall Symposium, 2011; see also N. Humphrey, *A History of the Mind* (Simon and Schuster, 1992) for a related view.
7. Perlis, "Five Dimensions of Reasoning in the Wild."
8. See P. Boltuc, "Mary's Acquaintance," *APA Newsletter on Philosophy and Computers* (Fall 2014): 25–31; and D. Perlis, "Consciousness and Complexity," *Annals of Mathematics and Artificial Intelligence* 14 (1995): 309–21, for related views.
9. N. Block, "On a Confusion about a Function of Consciousness," *Behavioral and Brain Sciences* 18 (1995): 227–87.
10. See N. Humphrey, "A Riddle Written on the Brain," *Journal of Consciousness Studies* 23, nos. 7-8 (2016): 278–87, for a similar view; but compare P. Boltuc, "The Philosophical Issue in Machine Consciousness," *International Journal on Machine Consciousness* 1, no. 1 (2009): 155–76.
11. D. Davidson, "Knowing One's Own Mind," *Proceedings and Addresses of the American Philosophical Association* 60 (1987): 441–58.
12. B. Libet, C. A. Gleason, E. W. Wright, and D. K. Pearl, "Time of Conscious Intention to Act in Relation to Onset of Cerebral Activity (Readiness-Potential). The Unconscious Initiation of a Freely Voluntary Act," *Brain* 106 (1983): 623–42.
13. J. Brody, M. Cox, and D. Perlis, "The Processual Self as Cognitive Unifier," *Proceedings*, Annual Meeting of the International Association for Computing and Philosophy, 2013.

REFERENCES

Block, N. "On a Confusion about a Function of Consciousness." *Behavioral and Brain Sciences* 18 (1995): 227–87.

Boltuc, P. "The Philosophical Issue in Machine Consciousness." *International Journal on Machine Consciousness* 1, no. 1 (2009): 155–76.

Boltuc, P. "Mary's Acquaintance." *APA Newsletter on Philosophy and Computers* (Fall 2014): 25–31.

Brody, J., M. Cox, and D. Perlis. "The Processual Self as Cognitive Unifier." *Proceedings*, Annual Meeting of the International Association for Computing and Philosophy, 2013.

Brody, J., D. Perlis, and J. Shamwell. "Who's Talking—Efference Copy and a Robot's Sense of Agency." AAAI Fall Symposium, 2015.

Chalmers, D. *The Conscious Mind*. Oxford University Press, 1996.

Davidson, D. "Knowing One's Own Mind." *Proceedings and Addresses of the American Philosophical Association* 60 (1987): 441–58.

Gettier, E. "Is Justified True Belief Knowledge?" *Analysis* 23 (1963): 121–23.

Humphrey, N. *A History of the Mind*. Simon and Schuster, 1992.

Humphrey, N. "A Riddle Written on the Brain." *Journal of Consciousness Studies* 23, nos. 7-8 (2016): 278–87.

Jackson, F. "Epiphenomenal Qualia." *Philosophical Quarterly* 32 (1982): 127–36.

Kripke, S. *Naming and Necessity*. Harvard University Press, 1980.

Libet, B., C. A. Gleason, E. W. Wright, and D. K. Pearl. "Time of Conscious Intention to Act in Relation to Onset of Cerebral Activity (Readiness-Potential). The Unconscious Initiation of a Freely Voluntary Act." *Brain* 106 (1983): 623–42.

Perlis, D. "Consciousness and Complexity." *Annals of Mathematics and Artificial Intelligence* 14 (1995): 309–21.

Perlis, D. "Consciousness as Self-Function." *Journal of Consciousness Studies* 4, nos. 4-5 (1997): 509–25.

Perlis, D. "Five Dimensions of Reasoning in the Wild." *AAAI*, 2016.

Putnam, H. "Meaning and Reference." *Journal of Philosophy* 70, no. 19 (1973): 699–711.

Putnam, H. "Models and Reality." *Journal of Symbolic Logic* 45, no. 3 (1980): 464–82.

Searle, J. "Minds, Brains, and Programs." *Behavioral and Brain Sciences* 3, no. 3 (1980): 417–57.

Winston, P. "The Strong Story Hypothesis and the Directed Perception Hypothesis." AAAI Fall Symposium, 2011.

CALL FOR PAPERS

It is our pleasure to invite all potential authors to submit to the *APA Newsletter on Philosophy and Computers*. Committee members have priority since this is the newsletter of the committee, but anyone is encouraged to submit. We publish papers that tie in philosophy and computer science or some aspect of "computers"; hence, we do not publish articles in other sub-disciplines of philosophy. All papers will be reviewed, but only a small group can be published.

The area of philosophy and computers lies among a number of professional disciplines (such as philosophy, cognitive science, computer science). We try not to impose writing guidelines of one discipline, but consistency of references is required for publication and should follow the *Chicago Manual of Style*. Inquiries should be addressed to the editor, Dr. Peter Boltuc, at pboltu@sgh.waw.pl