The Indeterminacy of Computation[α]

Fresco, Nir[Ψ], Copeland, B. Jack, and Wolf, Marty J.

Abstract. Do the dynamics of a physical system determine what function the system computes? Except in special cases, the answer is no: it is often *indeterminate* what function a given physical system computes. Accordingly, care should be taken when the question 'What does a particular neural system do?' is answered by hypothesising that the system computes a *particular* function. The phenomenon of the indeterminacy of computation has important implications for the development of computational explanations of biological systems. Additionally, the phenomenon lends some support to the idea that a single neural structure may perform multiple cognitive functions, each subserved by a different computation. We provide an overarching conceptual framework in order to further the philosophical debate on the nature of computational indeterminacy and computational explanation.

1. Introduction

Computational explanations are common in the study of cognition. Researchers regularly confront the question: 'What does a particular brain region or neural structure do and how does it do it?'. Answers often hypothesise that the region or structure computes some specific function. However, as we shall show, it is often *indeterminate* what function (understood broadly as a set of ordered pairs of elements) is computed by a given physical system—be it a Boolean gate, a single neurone, or a brain region. There may be no fact of the matter as to which of a number of different functions a specific physical system is computing.

[α] This is a pre-print of an article to be published in *Synthese*. The final version will be available here: https://link.springer.com/journal/11229/online-first

[Ψ] Corresponding author: nfresco@bgu.ac.il

This article clarifies and characterises the phenomenon of the indeterminacy of computation (Sections 2, 3 and 4); and articulates a methodological implication for the computational study of cognition, namely the need for an *extra explanatory step*. It also articulates a speculative implication concerning plasticity in the brain, namely that computational indeterminacy affords a mechanism for achieving neural plasticity. The first of these implications is the topic of Subsection 5.1 and the second of Subsections 5.2 and 5.3. We give a brief introduction here.

The methodological implication concerns the structure of computational explanations of biological systems. Suppose a scientist aims to explain the behaviour of an organism *O*; the investigation progresses and the observational data suggest that function *f* is computed by a neuronal structure *N* in *O*. It would, however, be hasty to conclude that the explanation of *O*'s behaviour is that *N* computes *f*, since *N* might indeterminately compute *f and* another function *g*. What is the scientist to do at this point?

An extra step, of one form or another, is required. One way forward would be to show that it is in the nature of *f* that, if *N* computes *f*, then *N* computes *f* determinately. Below, we sketch how this can be done in some discrete Boolean systems. However, it is also possible that the nature of *f* does not rule out indeterminacy. At this point, the scientist who wishes to hypothesise that the explanation of *O*'s behaviour is that *N* specifically computes *f* (and not some other function) must cast around for additional relevant empirical features of the situation—for example, the existence of adjacent neuronal structures with very specific properties. The scientist's aim will be to establish that these additional features rule out the hypothesis that it is the computation of *g*, or some other function that is different from *f*, that explains *O*'s behaviour. (While this additional investigation would undoubtedly require time and effort, it might lead to significant new hypotheses and discoveries about the functions of

adjacent structures.) We describe an example of such an investigation in Subsection 5.2, namely work done by Gabbiani and his colleagues on the locust.

The speculative implication concerning plasticity in the brain is as follows. A physical system that indeterminately computes a number of functions may be re-purposed to perform any of a range of distinct tasks (cognitive, perceptual, or other), with each task exploiting a different one of the computations that the system makes available. We speculate that the indeterminacy phenomenon may be leveraged in this way by resource-constrained neural systems. This hypothesis is responsive to, for example, Just and Varma's claim that each cortical area underpinning complex cognition can serve multiple cognitive purposes, thereby alleviating the limited neural resources of the brain (2007, 154).

Next, we illustrate the phenomenon of indeterminacy by means of a simple Boolean gate *G* with two input-channels and a single output-channel. *G*'s physical description is as shown in Table 1.[1] The concept of computational indeterminacy (although not the term) was present in the work of electronic engineer Ralph Slutz in about 1950 (Copeland, in progress). Slutz gave an example that is essentially similar to our gate *G*.

Input-channel 1	**Input-channel 2**	**Output-channel**
0.2–0.7 V	0.2–0.7 V	0.2–0.7 V
0.2–0.7 V	1.0–1.5 V	0.2–0.7 V
1.0–1.5 V	0.2–0.7 V	0.2–0.7 V
1.0–1.5 V	1.0–1.5 V	1.0–1.5 V

Table 1. Gate *G*'s output falls within the ranges shown in the 3rd column when its 2 inputs fall within the ranges shown in the 1st and 2nd columns.

[1] In Papayannopoulos et al. (in progress) it is argued that this indeterminacy arises from logico-mathematical interpretation of physical (or, possibly, abstract) states.

Gate *G* computes Boolean conjunction, since, if the voltage range 1.0–1.5 V corresponds to True and 0.2–0.7 V to False, Table 1 becomes the standard truth-table for conjunction. However, if the voltage range 0.2–0.7 V corresponds to True and 1.0–1.5 V to False, the gate computes Boolean inclusive disjunction. This is a simple example of the indeterminacy of computation. Later in this article, we generalise these considerations to non-Boolean systems.

Various forms and flavours of the phenomenon that we call the indeterminacy of computation (Fresco et al. 2016) have been discussed by diverse authors—pre-eminently Shagrir (2001, 2020)—and also (in chronological order) Dennett (1978, 2013), Block (1990), Sorensen (1999), Bishop (2009), Sprevak (2010), Fresco (2015), Piccinini (2015), Coelho Mollo (2017), and Dewhurst (2018). Some, for example Shagrir (2020), Bishop (2009) and Sprevak (2010), appeal to semantic features of the system to render it determinate what computation is being performed. However, these semantic features presuppose some theory of representation, and the problem of clarifying the nature of representation is a vexed one.

For that reason, many have resisted analysing physical computation in terms of representation (see, e.g., Dewhurst 2018; Fresco 2010; Miłkowski 2013; Piccinini 2015). Dewhurst, for one, appeals to *narrow physical properties* of the system, such as its specific input and output voltages, in order to avoid indeterminacy (2018). However, his proposal has some implausible consequences—for instance, an electrical AND-gate and a hydraulic AND-gate do not compute the same function (cf. Fresco & Miłkowski 2019).[2] The problem persists even when only electrical systems are considered: AND-gates operating within different voltage ranges—say one operating in the 0–10 V range and another operating in the 50–100 V range—do not compute the same function. Worse still, as we explain in Subsection 2.1,

[2] For a critical assessment of Dewhurst's proposal, see Coelho Mollo (2017). The latter introduces a distinction between 'computational equivalence' and 'equivalent insofar as computational individuation is concerned'.

Dewhurst's proposal entails the existence of AND-gates and OR-gates that compute the same function.

The present approach differs from earlier work in several major respects. We emphasise three. First, we do *not* regard indeterminacy as a *problem*, requiring a solution. Second, indeterminacy, far from being pernicious, is potentially *useful*. Third, we supply a general unified framework for describing and analysing the indeterminacy of computation. We develop this new conceptual framework in Section 2, drawing on fundamental results in Boolean logic. In Sections 3 and 5, we discuss in detail some implications of indeterminacy for philosophy of mind and cognitive science, and in Section 4 we consider some potential objections to our account. Then in Subsection 5.3 we introduce our *Massive Multiple Specifiability Hypothesis*, which speculates that the brain makes use of indeterminacy to secure plasticity of neuronal structures.

2. Conceptual Foundations: A Framework for Computational Indeterminacy

In this section, we develop a new conceptual framework for describing the indeterminacy of computation. Central to this framework is the concept we term *multiple specifiability* (we call the framework the '*MS*-framework'). Multiple specifiability is a cousin of the well-known concept of multiple realisability. We illustrate the *MS*-framework initially by means of Boolean logic, and we explain the phenomenon of Boolean duality, which is a basis for indeterminacy in discrete systems. Later in the article, we employ the *MS*-framework to describe and analyse indeterminacy in a broader range of cases.

2.1 Grounding functions and multiple specifiability

Underlying the description of any given physical system *S* as a computational system is what we call a *grounding function*.

Definition 1. A grounding function of a physical computational system *S* is a listing of tuples of physical inputs to *S* and physical outputs from *S*.

Grounding functions are functions from physical quantities to physical quantities (or from tuples of physical quantities to tuples of physical quantities), as inputs and outputs are always physical quantities. The system's physical inputs and outputs might be, for example, analogue voltages, numbers of water droplets, or light intensities. Grounding functions are typically expressed as mathematical equations of some form, either linear equations or more complex forms of equation. One and the same physical system may have a number of different grounding functions, corresponding to different levels of 'grain', such as macroscopic, microscopic, or quantum.

Notice that one is not free to specify a system's grounding function(s) *ad libitum*, since one is then on the high road to the implausible view that every physical system computes every computable function. (For discussions of this view, see, e.g., Searle (1980), Putnam (1988), Copeland (1996), Chalmers (1996), Piccinini (2012).) To select a grounding function is to propose a scientific hypothesis concerning the system, and this will be subject to the usual constraints governing scientific theorising. If *S*'s grounding function could be chosen arbitrarily, then *S* could support the computation of any Turing-computable function—but selection of the grounding function is never an arbitrary choice.

Table 1 provides an example of a grounding function, namely, gate *G*'s grounding function, at the level of grain and precision dictated by the type of voltmeter found in a standard electrical laboratory.

In describing Boolean gates such as *G*, electrical engineers typically employ a notation that suppresses some detail: inputs and outputs are assigned simple labels, e.g., '1' and '0'. We call the result of re-expressing a grounding function in this way a *labelling*.

Definition 2. A labelling of a grounding function is a function resulting from consistently assigning labels to the physical quantities in the grounding function.

A simple illustration of this definition is furnished by assigning labels to the voltages in *G*'s grounding function in the following way: a voltage is labelled '1' if and only if it is high (i.e., in the range 1.0–1.5 V in our example) and is labelled '0' if and only if it is low (i.e., in the range 0.2–0.7 V in our example). Assigning labels in this way produces the labelling g_1: {(0,0,0), (0,1,0), (1,0,0), (1,1,1)}. Labellings are a bridge between physical properties (e.g., voltages) and a mathematical system (e.g., Boolean logic). The labels we use in this article are always symbolic entities (such as '1', '0', 'T', 'TRUE', 'H', 'L', '1.444'). The immense utility of labellings lies in the fact that functions like g_1 are multiply realisable: numerous physical gates—of differing constructions and operating in very different voltage ranges—are described by g_1. Indeed, g_1 may also describe a gate whose inputs and outputs are water pressures or light intensities.

A more complicated example of a labelling is described in Subsection 4.4, where the states in a certain system's grounding function are assigned three distinct labels, depending on whether a state's magnitude is counted as low, medium, or high. In this case, a labelling of the grounding function is a set of triples of three labels.

We turn next to what we call *multiple specifiability*, which is, in a sense, an inversion of the well-known concept of multiple realisability. (In the following definition, we describe two functions as logically non-equivalent if there exists an argument for which they produce different values.)

Definition 3. A physical system S (e.g., a digital gate) is *multiply specifiable* if it has a grounding function that possesses at least two logically non-equivalent labellings (when using the same labels).

Definition 3 is illustrated by gate G. The two labellings g_1 and g_2: {(0,0,0), (0,1,1), (1,0,1), (1,1,1)} are inter-translatable by uniformly replacing '0' by '1' and '1' by '0'. However, g_1 and g_2 are not logically equivalent, since g_1 maps the truth-value pair (0, 1) to 0 while g_2 maps the same pair to 1. Thus, G is multiply specifiable. The upshot is that it is indeterminate what G computes. Since neither g_1 nor g_2 is privileged as a description of G's behaviour, it is indeterminate whether G computes conjunction (g_1) or inclusive disjunction (g_2). The same is true of any other physical system with labellings g_1 and g_2.

Our discussion of G generalises: *for any multiply specifiable computational system* (not necessarily a system limited to two effective states), *it is indeterminate what that system computes*. Nevertheless, multiple specifiability is by no means universal. Shortly we will illustrate this in the Boolean realm. But first, we quickly review the well-known Boolean concepts of duality and self-duality, as they are important for our discussion.

2.2. Boolean duality

Conjunction {(0,0,0), (0,1,0), (1,0,0), (1,1,1)} and inclusive disjunction {(0,0,0), (0,1,1), (1,0,1), (1,1,1)} are examples of dual functions, as are exclusive disjunction {(0,0,0), (0,1,1), (1,0,1), (1,1,0)} and logical equivalence {(0,0,1), (0,1,0), (1,0,0), (1,1,1)}. A Boolean function g is here defined as the dual of a different Boolean function f if and only if uniformly

exchanging 1s and 0s in *f* produces *g*, as in the two examples given above.[3] [For a formal definition of duality, see Crama and Hammer (2011, 13).]

Some Boolean functions have the special property that applying the above process of exchanging truth-values simply produces the very *same* function. An example is Boolean negation {(0,1), (1,0)}. Functions with this property are called *self-duals*.

Although multiple specifiability is very common in the Boolean realm, it is far from universal. Multiple specifiability—and, therefore, indeterminacy—does not arise in the case of circuits that compute self-dual functions. Any Boolean circuit built exclusively from components that compute self-dual functions must itself compute a self-dual function[4], and thus is not multiply specifiable. Nevertheless, there are cases in which the inclusion of a single multiply specifiable gate results in the overall circuit also being multiply specifiable.

We turn next to the relevance of our analysis for the study of cognition.

3. The Significance of Indeterminacy for the Study of Cognition

Indeterminacy of computation extends to the cognitive realm. Neural 'hardware', where the grounding function is expressed in terms of electrochemical signals (e.g., voltages) or neurotransmitting chemicals (e.g., concentrations), may be indeterminately computing many functions. In this section and in Section 5 we explain in detail how the indeterminacy phenomenon may impinge on cognition, and how the computational brain may exploit it. We first review the debate between the literalist (or realist) and non-literalist (or instrumentalist)

[3] Typically, the requirement that *f* and *g* be different functions is not included when defining duality; we enter this requirement here to simplify the following discussion.

[4] This is a theorem, and it is provable by iterating the following inductive process in Boolean logic. Let $\vec{x}$ and $\vec{y}$ be two strings of input variables, and let z be a single input variable. If $f(\vec{x}, z)$ and $g(\vec{y})$ are self-dual functions, then $f(\vec{x}, g(\vec{y}))$ is self-dual even when some of the variables amongst $\vec{x}$, $\vec{y}$, z are not distinct. [For proof of this second theorem, see Crama and Hammer (2011, 173–174).] Notice that our theorem gives a sufficient, but not a necessary, condition for a complex Boolean function to be self-dual (there are self-dual functions some of whose components are *not* self-dual).

views of neural computation, arguing that the phenomenon of indeterminacy impacts importantly upon computational cognitive theories (Subsection 3.1). Second, because neural systems are typically viewed as continuous rather than discrete, we discuss the indeterminacy of computation in continuous systems (Subsection 3.2).

We make two preliminary observations. First, this article takes no stand on whether the computational paradigm in neuroscience is correct or profitable. There are certainly non-computational approaches to understanding the brain, just as there are non-computational approaches to understanding the functioning of other biological organs, such as the heart and the liver. Precursors of non-computational approaches to understanding the brain were being pursued by neuroscientists long before the work of Turing, McCulloch, Pitts, von Neumann, and others ushered the computational paradigm into neuroscience. The form of our discussion is this: *if* a neuroscientist adopts the computational paradigm, then they must confront the indeterminacy of computation. (Likewise, *computational* approaches to understanding the functioning of hearts and livers [e.g., Cvitanović et al. 2017] must also confront the indeterminacy of computation.)

Second, we mention a possible response from computationalists to the explanation issue which we consider inadequate, namely, that in light of the indeterminacy phenomenon, one should simply aim to give explanations at a higher level of abstraction. Accordingly, rather than an explanation at the Boolean-function level, one might explain the behaviour of a system by saying simply that the system computes one or other of two dual functions. For example, in the case of the duals AND and (inclusive) OR, the explanation would be simply that the system computes *either* AND *or* OR. Call this the *con-dis* explanation. The con-dis explanation is still predictive of the system's behaviour, but it is important to note that it is not as informative as the conjunction hypothesis or the disjunction hypothesis in isolation. As our

case study in Subsection 5.2 will illustrate, to rest with an explanation like con-dis would be to curtail the empirical enquiry prematurely: more can be done.

3.1 Computational literalism versus computational non-literalism

In the introduction to his classic book *Biophysics of Computation*, Koch stated that '[t]he brain computes' is 'accepted as a truism by the majority of neuroscientists' (1999, 1). Already in the 1960s, retinal ganglion cells in cats were described as adding, i.e., computing: '[S]ignals from photoreceptors in both the centre and surround summating regions of an X-cell receptive field add linearly before they affect the discharge of the ganglion cell' (Enroth-Cugell & Robson, 1966, 545). More recent examples abound. Primary visual cortex (V1) cells in monkeys are described as computing a sum: '[V1] cells compute a linear sum of the responses of lateral geniculate nucleus … neurons' (Carandini & Heeger, 1994, 1333). The locust's LGMD neurone is described as computing in the course of generating escape behaviours (Gabbiani et al., 2002), an example we discuss in more detail in Subsection 5.2.

On the Kochian view, the brain literally computes: computation is not merely a metaphor for describing the brain's activity. According to this literalist view, the brain receives sensory inputs, encodes them into various biophysical variables (e.g., neuronal firing rates), and then computes over these variables. Some would say that Koch is overstating matters in his claim that this view is widely accepted in the neurosciences. There is, in fact, controversy about whether computational neuroscience is committed to the literalist view that the brain *is* an implemented computational system (Fresco, 2014). There is also considerable controversy about whether neural systems compute at all in any robust sense, and, indeed, about what it even means to say that a neural system *computes*. (According to Piccinini and Bahar, neural computation is neither digital nor analogue, but *sui generis* (2013).) In view of such fundamental disagreements, we will discuss the distinction between literal and non-literal

computational explanations in the cognitive sciences a little further, and also the relation of the indeterminacy phenomenon to literal and non-literal computational explanations.

In a statement of literalism, neuroscientists Carandini and Heeger asserted that although 'it is unlikely that a single mechanistic explanation will hold across all systems and species', nevertheless 'what seems to be common is not necessarily the biophysical mechanism but rather the computation' (2012, 58). In their view, the brain, in solving different cognitive problems, relies on a set of canonical neural computations that repeat across brain regions and modalities. Another vivid statement of computational literalism is provided by neuroscientists Knill and Pouget, who hypothesise: 'The brain represents information probabilistically, by coding and computing with probability density functions' (2004, 713). The debate about computational literalism is ongoing in the cognitive sciences, and it is by no means clear which side, literalism or non-literalism, will emerge as victorious.

The distinction between computational literalism and computational non-literalism is also evident in the field of cognitive modelling. [As Busemeyer and Diederich reported, 'More than 80% of the articles appearing in major theoretical journals of cognitive science involve cognitive modeling' (2010, 1).] Many computational cognitive models—but certainly not all—are intended to be understood non-literally: they aim to predict brain function without commitment to the neural reality of computational mechanisms.

Regardless of whether computational explanations or models take a literalist or non-literalist form, the indeterminacy of computation may arise. It is clear that on the literalist view, indeterminacy can arise in brain computations, given arguments that will follow (Subsections 3.2, 4.2, 4.3, 5.1 and 5.2). The case of non-literalism subdivides. A non-literalist may locate the computations employed in the model *entirely within the model*; for such a non-literalist, computational neuroscience is no more about computations in the brain than computational

cosmology is about computations in galaxies. On the other hand, a non-literalist may regard computation in the brain as a *useful fiction*, and here the indeterminacy of computation still intrudes. Whether we are working inside or outside an in-the-fiction operator, the same considerations apply. For example, suppose it is part of the fiction that the cerebellum consists of Boolean circuits of the type described in Section 2. In this particular fiction, a labelling of the grounding function of these cerebellar circuits is {(0,0,0), (0,1,0), (1,0,0), (1,1,1)}. For the reasons given in Section 2, it is indeterminate, in the instrumentalist's fiction, whether these circuits compute conjunction or (inclusive) disjunction. Thus, the non-literalist, just as much as the literalist, is confronted by the need for an extra explanatory step, as explained in Section 1 (and see further Subsection 5.1).[5]

Indeterminacy, like inconsistency, cannot be trivially accommodated by non-literalism. The existence of inconsistencies within the scope of an in-the-fiction operator is widely acknowledged to be a serious problem in the semantics of fiction, and the problem has generated a large technical literature (e.g., Badura and Berto 2019; Frigg 2010; Lewis 1978, 1983). There is certainly no easy fix. The non-literalist cannot, for example, simply *stipulate* which part of a contradiction within a fiction is true in the fiction and which is false (Lewis 1983, Proudfoot 2006). The non-literalist needs to carry out some challenging theoretical work to accommodate the possibility of inconsistency within the fiction. Similarly, indeterminacy within the scope of an in-the-fiction operator is an important and difficult problem. In the case of an indeterminate computation, the non-literalist cannot simply stipulate which of the

[5] Someone might suggest that any description in Boolean terms of a specific circuit in the brain, such as that the neural circuit in question performs conjunction, is from an *external* point of view, whereas *intrinsically* the circuit simply grows and behaves subject to the relevant laws of biochemistry and electrodynamics. If any computationalist agrees with such a suggestion, then presumably they must be a non-literalist — and if they favour, say, a conjunction-based description of the circuit, then, just as much as the literalist, they need to provide a *reason* for preferring this description to one couched in terms of the dual function, inclusive disjunction. See further Subsection 5.1.

competing functions is to be privileged in the fiction—given that other statements in the fiction entail the existence of indeterminacy in the fiction (in the way described above). Again, the non-literalist is required to undertake some challenging theoretical work in order to accommodate indeterminacy.

To highlight the moral of this subsection: all realist computational cognitive theories and models, and all computational cognitive models taking the fictionalist stance just outlined, are potentially affected by the indeterminacy phenomenon. This impact is overall positive, as we explain in Subsections 5.2 and 5.3.

3.2 Indeterminacy in continuous systems

The systems we have discussed so far, such as gate *G*, produce 'stepped' output (e.g., *G*'s output lies either in the range 0.2–0.7 V or a 'step up' in the range 1.0–1.5 V). The systems' inputs are also stepped. Other systems of interest, such as neurones, have outputs and inputs that are not stepped but vary more smoothly. These are called *continuous* systems. Continuous systems are also subject to computational indeterminacy.

To illustrate this claim, we consider a black box: the box has two input-channels and one output-channel. These channels carry continuous physical quantities γ, η, and λ (e.g., voltages). Schematically, the equation describing the box's relevant physical behaviour (the grounding function) is:

$$g(\gamma, \eta) = \lambda,$$

where λ is carried by the output-channel, and γ and η, respectively, by the input-channels.

A labelling of the box's grounding function employs labels for arbitrarily small adjacent segments of the box's inputs and outputs. Conveniently, decimal numerals may be used as the labels (the more significant figures there are in the numerals, the smaller the segments considered). g_L is a labelling of the box's grounding function employing a (positive) decimal

numeral running to, say, four decimal places as a label for each segment of γ, and each segment of η and λ. We write this function as $g_L(i,j) = x$.

Suppose further investigation of the box's behaviour reveals that $x = i + j$ (i.e., numeral x denotes the number resulting from adding the number denoted by i to the number denoted by j). That is to say, the grounding function can be relabelled, producing g_L', by replacing the labels of segments of output by labels consisting of '+' flanked by the labels of the corresponding segments of the two inputs. We write this new function: $g_L'(i,j) = i + j$.

There is yet another way of labelling the grounding function, producing $g_L''(i,j) = i \times j$. This is because the addition of logarithms amounts to the multiplication of the corresponding anti-logarithms (exponents). If the decimals labelling the input segments are logarithms, then the box multiplies two numbers; the product is the antilog of the corresponding output segment's label.

Since g_L' and g_L'' are non-equivalent labellings of g_L, it follows by Def. 2 that the black box is multiply specifiable.[6]

When the black box is considered in isolation, there is no reason to prefer either description, g_L' or g_L'', over the other: the box is multiply specifiable and it is indeterminate which computation is performed. If anything in the encompassing system settles whether the inputs and outputs are logarithms, this lies beyond the black box. (Notice that this indeterminacy, which is akin to the AND/OR case above, has nothing to do with the fact that, in continuous systems, different inputs may be so close to one another that it is difficult or impossible to distinguish the exact function that the system is computing.)

[6] The requirement in Def. 3 that the same labels must be used is satisfied, since × may be expressed equivalently in terms of repeated addition.

This hypothetical example illustrates that multiple specifiability arises also in continuous systems. Real neural systems also may exhibit multiple specifiability. Like our black box, the soma of the LGMD neurone is multiply specifiable. According to one specification, the LGMD soma adds, and according to the other, the soma multiplies: it multiplies by adding logarithms. We will return to this interesting example in Subsection 5.2.

The indeterminacy of computation is, then, by no means limited to Boolean and other discrete systems.

4 Anticipating Some Objections

In this section, we consider four very different objections to our analysis thus far. The first is that the computation performed by our hypothetical black box is actually *not* indeterminate. The second objection suggests that neural systems are not vulnerable to computational indeterminacy, since they operate in noisy, stochastic environments. The third maintains that computational indeterminacy is confined to relatively small, and perhaps uninteresting, systems. The fourth considers a new case of computational indeterminacy, discussed in Shagrir (2020)—the objection is that this is a different kind of indeterminacy, not covered by the allegedly general framework introduced in Section 2.

4.1 Is there really indeterminacy in the black box?

A possible objection to the discussion in Subsection 3.2 is that the black box is far from analogous to gate *G* and does *not* exhibit indeterminacy. The box computes *only* addition (the objection goes): whatever physical magnitudes the box receives as inputs, it simply sums them up; for example, inputs of 1 V and 2 V produce 3 V as output.

This objection, however, confuses the physical accumulation of voltages with addition over the numbers. Moreover, it misses the point. The box is able to perform *both* computations: it

adds and it multiplies. As we have seen, the grounding function certainly can be labelled in such a way that the box computes addition; but the grounding function can equally be labelled so that the box computes multiplication.

4.2 Noise

Another potential objection to our claim that computational theories of cognition are impacted by the phenomenon of indeterminacy runs as follows. Neural systems operate in noisy, stochastic environments on continuous quantities, which may be grossly distorted by saturation and threshold non-linearities. Indeterminacy does not arise in noisy biological systems since—according to the objection—noise somehow cancels indeterminacy out.

Our response is threefold. First, indeterminacy of computation may exist in systems that operate on continuous quantities, as previously argued. Second, single neurones that compute—possibly nonlinear—threshold functions on potentially thousands of inputs may still be classified fundamentally as interconnected Boolean units that either fire or not, depending on whether some threshold has been exceeded. This accords with the all-or-none principle of neuronal firing, which states that action potentials produced in a neurone all reach the same maximum value: they are either produced as a whole or not at all.

Third, indeterminacy is not cancelled out by noise. Noise can in fact lead to a form of indeterminacy even more radical than that discussed here. In the case of gate *G*, for example, it is at least determinate how many input-channels there are, but for a system existing in a noisy real-world environment, even that much may not be determinate. As Dennett pointed out, the straightforward physical facts do not determine whether impinging variables such as 'changes in temperature or relative humidity, or sudden accelerations' count as signal or interference, i.e., signal or noise (Dennett 1978, 258). In the case of an artefact, he said, what 'counts as interference, and what as a physical change "read" as input by the machine is relative to the

designer's choice' (ibid). However, in the case of brains (or smaller neuronal structures) there is 'no Designer to interview, no blueprint to consult' (Dennett 1978, 261). Dennett concluded that one 'complicated chunk of the world' can be many different Turing machines at once.

4.3 Complexity and indeterminacy

One might perhaps think that the probability of indeterminacy decreases as the size of a computational system grows, giving rise to the objection that the phenomenon of indeterminacy is limited to relatively small computational systems of no great interest.

The short answer to this objection is that whether a computational system is subject to indeterminacy depends not on its size but on which function(s) it computes. In the Boolean case, at least, indeterminacy—far from becoming rarer—becomes increasingly common as functions with ever larger numbers of inputs are considered. This is expressed in the following theorem.

Theorem 1. The probability that a Boolean function with n inputs is self-dual is p_n, where $p_n = 2^{2^{n-1}} / 2^{2^n}$.

The theorem follows from the fact that there are 2^{2^n} Boolean functions on n variables but $2^{2^{n-1}}$ self-dual Boolean functions on n variables.[7] To illustrate the impact of the difference between the terms 2^{2^n} and $2^{2^{n-1}}$, consider the probability p_5 that a five-input Boolean function is not self-dual (recall that non self-dual Boolean functions give rise to indeterminacy). That probability is $1 - \frac{2^{2^4}}{2^{2^5}} \approx 0.99$.

We certainly have not shown in general that the probability of indeterminacy increases with the number of a function's inputs. However, we have shown this in the Boolean case, thus

[7] See, e.g., Crama and Hammer (2011, 173–174); the fact just stated is a corollary of their Theorem 4.14.

shifting the burden of proof onto any objector who wishes to maintain that, nevertheless, there are functions of interest to cognitive scientists for which this is not the case.

Pressure on the objector increases if one considers, not only Boolean functions per se, but also circuits that compute them. McCulloch and Pitts (1943) made the assumption that the brain is essentially a Boolean system. Let us follow them in that assumption for a moment and look at its consequences for the objection being considered. We may assume that the number of input-channels into a Boolean neuronal structure is indicative at least of the order of magnitude of the number of inputs into the structure's grounding function, and so is indicative of the order of magnitude of the number of inputs into the Boolean function or functions that the structure computes. The fact that a typical neurone has several thousand inputs—coupled with the assumption just stated—indicates that the a priori probability of a many-neurone structure being multiply specifiable is generally high in a McCulloch-Pitts type of model.

The availability of an analogue of Theorem 1 for circuits, rather than functions, would place further pressure on the objection under consideration. However, the step from functions to circuits is by no means a straightforward one, since, for any function, there are many different circuits that compute it. (To give a trivial example, if circuit C computes function *f*, then circuit CNN also computes *f*, where CNN represents the result of attaching two negators serially to the output-channel of C at which the values of *f* appear.) Moreover, some circuits compute many different functions. For instance, Turing showed that there exists a circuit consisting only of NAND gates that implements 'a universal [Turing] machine with a given storage capacity' (1948, 422). This 'semi-universal' circuit can compute every Boolean function that is able to be computed without exceeding the given storage capacity.

In order to prove an analogue of Theorem 1 involving circuits, we focus on a constrained subset of Boolean circuits, which we call *characteristic* circuits. χ is a characteristic circuit for

a Boolean function *f* if, and only if, (i) χ computes *f*, and (ii) χ computes no other functions except logical equivalents of *f* and the dual of every function computed by χ. Every Boolean function has at least one characteristic circuit. We say that a collection of characteristic circuits is *n-complete* if, and only if, (i) every circuit in the collection computes some Boolean function on *n* variables, and (ii) every Boolean function on *n* variables is computed by exactly one of the circuits in the collection.

Theorem 2. For every *n*-complete collection of characteristic circuits, the probability that any circuit from the collection is multiply specifiable is π_n, where $\pi_n = 1 - (2^{2^{n-1}} / (2^{2^n-1} + 2^{2^{n-1}-1}))$.

This is because the size of any *n*-complete collection of characteristic circuits C_n is $2^{2^{n-1}} + 2^{2^{n-1}-1}$, while the number of circuits in C_n that compute self-dual functions—that is, the number of circuits in C_n that are *not* multiply specifiable—is $2^{2^{n-1}}$.

Clearly π_n, the probability of indeterminacy, increases as *n*, the number of inputs, grows. Of course, we have not shown this for all collections of circuits, and not even for all collections of Boolean circuits. Nevertheless, this result again places the burden of proof upon anyone who suggests that there are classes of functions of interest to cognitive scientists where the probability of the occurrence of indeterminacy *decreases* as *n* grows.

4.4 Another kind of computational indeterminacy?

According to this objection, the theoretical framework set out in Section 2 is incomplete. There is, the objection maintains, a further kind of computational indeterminacy, related to but different from the computational indeterminacy described in Section 2, and this new kind of computational indeterminacy is not accommodated by the *MS*-framework. The indeterminacy

in question results from *grouping* the same physical properties of a system *differently*. The example we will use to illustrate this idea is adapted from Shagrir (2020).

S is a two-input-channel, single-output-channel system, whose valid inputs and outputs fall within three voltage ranges: (1–3 V), (3–5 V), and (6–10 V). *S* outputs a voltage in the high range (6–10 V) when it receives input of two voltages also in the high range; and it outputs a voltage in the low range (1–3 V) when it receives two voltages in the low range; and under all other input conditions it outputs a voltage in the mid range (3–5 V). *S* is a three-state system, in the sense that it responds differentially to voltages in three different ranges.

S's grounding function can be labelled as follows (where 'L' labels voltages in the low range, 'M' in the mid range, and 'H' in the high range) *k*: {(L,L,L), (L,M,M), (L,H,M), (M,L,M), (M,M,M), (M,H,M), (H,L,M), (H,M,M), (H,H,H)}. Now, *k* can be relabelled in two non-equivalent ways. First, *k′*: {(0,0,0), (0,1,1), (1,0,1), (1,1,1)} is also a labelling, where '0' labels voltages in the low range and '1' labels voltages in either the mid range or the high range (duplicate triples are omitted). This is (inclusive) disjunction. Second, if '1' labels voltages in the high range and '0' labels voltages in the low and mid ranges, *k″*: {(0,0,0), (0,1,0), (1,0,0), (1,1,1)} is another labelling (again duplicate triples are omitted). This is conjunction.

Since the two labellings *k′* and *k″* are non-equivalent, and there is no reason to favour one over the other, *S* indeterminately computes two different Boolean functions, conjunction and (inclusive) disjunction. However, this indeterminacy does arise from the multiple specifiability of *S*. (*S* is multiply specifiable since its grounding function possesses at least two logically non-equivalent labellings employing the same labels, as required by Def 3.) *S*'s indeterminacy, then, is describable within the *MS*-framework (we have just done so).

The interesting question of whether *S*'s indeterminacy is, or is not, of a different kind from *G*'s remains open, though. *S*'s indeterminacy results from grouping the voltage ranges differently. In *k′*, the mid and high ranges are grouped together, whereas in *k″* it is the low and mid ranges that are grouped together. Under each grouping, *S* is a two-state system, but a different two-state system in each case. We will not pursue the same-or-different question here. The issue of whether this really is a new kind of indeterminacy is considered by Papayannopoulos et al. (in progress), Fresco (forthcoming) and Copeland (in progress). Our response to the objection is simply that, *even if* this is a distinct kind of computational indeterminacy, it can be accommodated in the *MS*-framework.

5 Indeterminacy and Computational Explanation in Neuroscience

In this section, we discuss the impact of the indeterminacy of computation on computational explanation in cognitive science. Subsection 5.1 outlines the problem. Subsection 5.2 describes a detailed case study: the lobula giant movement detector (LGMD) neurone of the locust. In Subsection 5.3, we propose a speculative hypothesis about massive multiple specifiability in the brain.

5.1 On the (in)completeness of computational explanation

Here we return to the issue of the *extra explanatory step*, broached in Section 1. To illustrate the difficulty, suppose *S* is multiply specifiable, as computing the function *f* and as computing a different function *g*. Why is the hypothesis that *S* is computing *f* preferable to the hypothesis that *S* is computing *g*? When a computational-level explanation identifies *S* as computing *f* rather than *g*, some good reason for preferring *f* is needed. In the absence of such a reason, the computational explanation is incomplete.

Computational explanations of the behaviour of simple two-input-channel McCulloch-Pitts style neurones with a labelling g_1 (or g_2) can be couched in terms of either conjunction or (inclusive) disjunction (see Subsection 2.1). There are likely to be multiple alternative explanations of a Boolean neurone's behaviour when the neurone has a large number of input-channels (and neurones in the brain may have thousands of input-channels). A scientist who favours one of these multiple explanations over the others needs a reason for doing so—and in producing this reason, the scientist clearly must cast the net beyond those aspects of the neurone's behaviour that are consistent with each and every one of its multiple specifications. The explanation must look *beyond* the neurone's immediate input–output behaviour, since this behaviour is what is expressed by the grounding function—and, *ex hypothesi*, the grounding function can be labelled in different ways, so producing the multiple specifiability and indeterminacy that leads to the multiple explanations.

Studies of the locust's LGMD neurone illustrate how this may be done in a complex, biologically realistic situation, as we will explain.

5.2 More on the LGMD neurone

In this section, we argue that the same kind of indeterminacy arising in the hypothetical black box (Subsection 3.2) also arises within a single neurone: the LGMD neurone. This neurone was first implicated in looming detection in 1977 (Schlotterer, 1977). The LGMD neurone is believed to track the kinematics of objects approaching on a collision course with the locust (Jones & Gabbiani, 2012). Its peak firing rate signals when the approaching object's subtended size reaches a specific angular threshold on the locust's retina, and this typically initiates the locust's escape behaviour.[8]

[8] Gabbiani and his colleagues explain that the LGMD neurone fires throughout object approach with a rate that increases as the object grows larger, peaks, and eventually decays as collision becomes imminent. The neurone,

We focus here on the multiple specifiabilty of the LGMD soma—or, more precisely, of what we call the neurone's *merge zone*. The merge zone is where the neurone's three dendritic subfields effectively merge (see Fig. 1). (In vertebrates, the soma is typically close to the spike initiation zone and is directly between the dendrites and axon. However, this is not the case in the locust neurone, where the soma itself is not on the electrical path of signalling, but on a side branch. For the sake of simplicity, we ignore the neuroanatomical details concerning where the actual processing of inputs into the LGMD neurone takes place and assume that it occurs at the merge zone.) The grounding function of the LGMD's merge zone is describable in terms of the intrinsic properties of the cell's membrane. As with the black box in Subsection 3.2, the grounding function may be labelled as simple addition (specifically, the addition of the approaching object's angular size θ, which is the inhibitory input, and its velocity ψ, which is the excitatory input); and the grounding function may also be labelled as multiplication (of θ by ψ).

arguably, computes the time at which a specific threshold angle is reached during object approach. For the peak firing rate and subsequent escape behaviours occur at a fixed delay (attributed to neuronal processing time) after the object has reached a constant angular size on the retina.

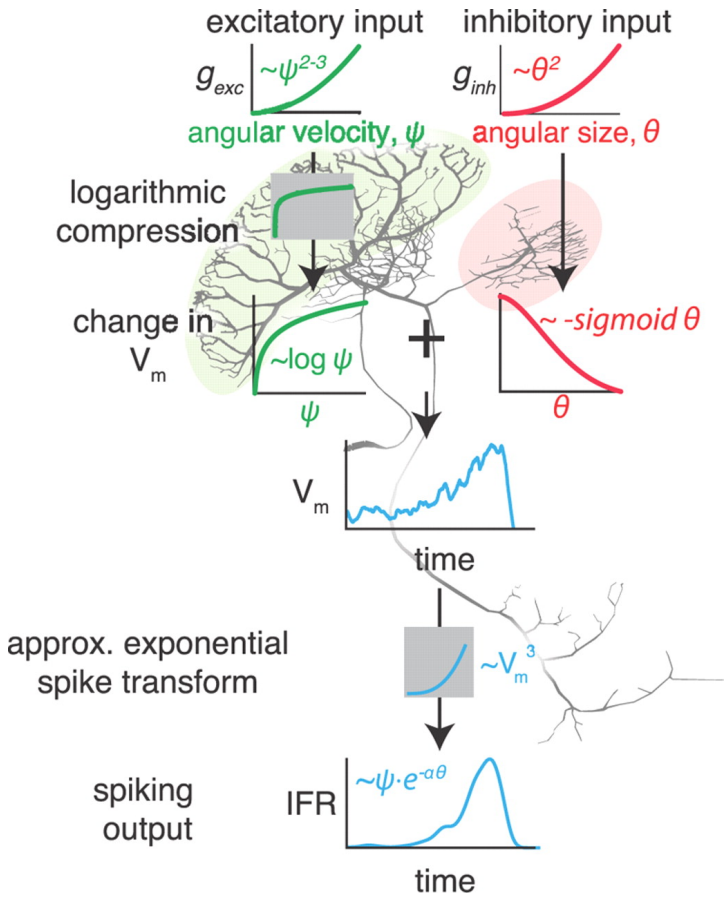

Figure 1. **Computation in the locust's LGMD neurone.** When faced with a looming stimulus simulating an object approaching on a collision course, excitatory inputs are received that are proportional to the object's angular velocity raised to a power of 2 to 3. The inhibitory inputs are approximately proportional to the square of the object's angular size. In this biophysical model, the excitatory signal undergoes a logarithmic compression in the excitatory dendritic subfield, while the effect of the inhibitory inputs on the mean resting membrane potential (V_m) results in a sigmoidal (nearly linear) dependence on angular size. The sum of these signals in the merge zone (the juncture immediately above the '+' sign) is then exponentiated in the axonal spike initiation zone, resulting in an output relationship that signals an angular threshold size through its firing rate peak. (The figure is reprinted from Jones and Gabbiani (2012, fig. 8) with permission from Gabbiani. Permission from the Journal of Neuroscience is pending.)

Gabbiani et al. initially considered two competing hypotheses about what the LGMD neurone *in toto* (not just the merge zone) computes in the specified context (stationary collision avoidance): namely, addition versus multiplication. However, they gathered extensive empirical data that favours the second alternative over the first, saying '[a]dditive

combinations of [θ and ψ] could not fit experimental firing rate profiles' (Gabbiani et al., 2002, 321). The results of their investigations of the multiplication hypothesis[9] were reported in a string of publications (e.g., Gabbiani et al., 2002; Gabbiani et al., 1999; Jones & Gabbiani, 2012). This research is a paradigmatic example of resolving a case of underdetermination of theory by data: competing hypotheses concerning a single neurone's computation were evaluated experimentally by means of a highly resourceful search for additional data.[10]

Their main reason for favouring that specific version of the multiplication hypothesis is as follows. Gabbiani et al. claim that the collected data 'support the hypothesis that … multiplication is achieved within the [LGMD] neuron through addition of [θ and ψ] transformed logarithmically, followed by an approximate exponentiation' (Jones & Gabbiani, 2012, 4923–4924). They identified empirically what we will call an *upstream mechanism* and a *downstream mechanism* whose presence constitutes strong evidence for the multiplication hypothesis. (The terms 'upstream' and 'downstream' are used here relative to the direction of information flow through the computing system; a downstream mechanism is situated subsequent to the system's output-channels, and an upstream mechanism is situated prior to the system's input-channels.)

In detail, the upstream and downstream mechanisms are localised within the LGMD neurone's dendrites and its axonal spike initiation zone. The former consists of inhibitory and excitatory dendritic subfields (see Fig. 1). Gabbiani and his colleagues report that the presynaptic inputs are logarithmically transformed within the dendritic subfields before being processed at the merge zone (assuming that it is indeed the merge zone that carries out the processing), and then exponentially transformed at the axonal spike initiation zone into the

[9] This particular version of the multiplication hypothesis is one of three they considered. They first advanced the multiplication hypothesis back in the 1990s (Hatsopoulos et al., 1995).

[10] Computational neuroscientists are typically well-aware of the fact that whatever specific function they focus on, the function will generally be underdetermined by the available empirical data (see, e.g., Gabbiani 1995).

LGMD neurone's firing output.[11] The upstream structure—the dendritic subfields—performs logarithmic compression on inputs to the LGMD, and the downstream structure—the axonal spike initiation zone—performs exponentiation on the membrane potential output.

Notice, however, that even once the *underdetermination* (of theory by data) is mitigated by means of the further empirical investigation just described, the *indeterminacy* (of computation in the merge zone) remains. The effect of the upstream and downstream structures is that one of the two functions computed by the LGMD merge zone is selected while the other is, so to speak, passed by. One of the functions—multiplication, in the case of collision aversion—is 'tapped off' the merge zone by the upstream–downstream milieu.

Might the other function (addition) conceivably be tapped off under different circumstances? In collision aversion, the inputs become related to angular size and velocity presynaptically to the LGMD. However, it is certainly conceivable that in some other context (i.e., other than collision aversion in stationary mode) the LGMD neurone *may*—as in the case of the hypothetical black box—perform simple addition of its inputs. Other computations occur throughout the locust's visual system, both pre- and post-synaptically to the LGMD neurone. In a different visual task, say in response to *receding* objects, the inputs may become related to the relevant visual properties differently. In such a context, addition, rather than multiplication, could be tapped off the merge zone.

Moreover, according to computational literalism, all this holds whether or not some external observer has actually introduced labellings of the grounding function. Mechanisms in the milieu of the locust's visual system could have been tapping off one or another—or both—of the available functions for millennia prior to the arrival of human theorists and their labels.

[11] For the sake of accuracy, we note that the inhibitory input (unlike the excitatory input, which was reported to be logarithmically compressed) was reported to be more subtly transformed, specifically, sigmoidally transformed (Jones & Gabbiani, 2012, 4931).

Here, then, is a general methodology for addressing questions of the type raised in Subsection 5.1. When there exist competing computational explanations of some neurone's (or neural circuit's) behaviour, researchers may discriminate empirically between them by searching for relevant upstream-downstream structures. This could be done in the way illustrated by Gabbiani et al.'s ongoing investigation of how the LGMD neurone implements multiplication. If an upstream–downstream milieu is identified that taps off one of the multiple functions, *f*, then the *f*-involving explanation is preferable. Once all relevant upstream–downstream structures have been identified, the computational explanation of the behaviour of the neurone (or circuit) may be regarded as complete.[12] The search for these upstream–downstream structures may lead to important new discoveries.[13]

5.3 Indeterminacy and plasticity

The indeterminacy of computation may play an important role in the plasticity of various neural structures in the brain. It may be exploited by the brain when limited computational

[12] At this point, a philosophical problem arises (see Shagrir 2020). Assuming that the upstream–downstream structures that have been identified are computational, the argument about indeterminacy can be reapplied to these structures. If it is indeterminate what these structures are computing, then it is indeterminate what the original neurone is computing. There are various ways in which this attempt to reapply the indeterminacy argument to the wider system may fail. For example, it will fail if the relevant upstream–downstream structures are not themselves computational in nature, or if the entire system consisting of the target system (e.g., a circuit, a neurone, or a merge zone) and its surrounding upstream–downstream milieu has the property of self-duality (extended appropriately to the neural context). In those cases where the argument can be validly reapplied, the search for upstream–downstream structures will arguably yield a relative result, rather than an absolute one. The isolation of a suitable milieu *M* surrounding the target system *S* may make it possible to say that *relative to M*'s *computing f*, *S* is computing *g*. It is conceivable that, in some empirical situations, such statements of relative computability are the best that researchers can hope to establish.

[13] Piccinini (2015, 16, 43) seems to appeal to what he calls the 'narrow context' of a computational system in order to determine what computation is being performed in that context. Our proposal might perhaps be regarded as a detailed working out of Piccinini's schematic suggestion. However, our proposal is in fact compatible with both *semantic* individuation of computation (e.g., Sprevak (2010) or Shagrir (2020)) and *mechanistic* individuation of computation (Piccinini 2015). Moreover, it is important to note that there are also key differences between our account and Piccinini's. For example, Piccinini seems to suggest that appealing to the system's narrow context *eliminates* the indeterminacy, while our suggestion is that appealing to the upstream–downstream milieu does not eliminate the indeterminacy of the LGMD merge zone's computation. A second upstream–downstream structure might simultaneously tap off a *different* computation from the merge zone. Another important difference is that Piccinini—unlike us—believes that, in 'naturally occurring' mechanisms, different tasks are usually 'subserved by a different process within the mechanism' (ibid, 16). See Fresco (forthcoming) for an alternative, *wide* mechanistic strategy of computational individuation.

resources are a constraining factor. The brain seems to (re)configure itself adaptively in response to varying processing demands. '[A] given neuronal structure can perform multiple functions that depend on the areas with which it interacts' (Price & Friston, 2005, 262). Multiple purposes served by a single structure may include, for example, regulatory, cognitive, and affective tasks (Menon & Uddin, 2010). '[R]ather than developing new structures de novo, resource constraints and efficiency considerations dictate that whenever possible neural … resources should have been reused and redeployed in support of any newly emerging cognitive capacities' (Anderson, 2014, 7). Evolutionary considerations also seem to point in this direction: the evolution of more complex brains from simpler ones is a process favourable to the deployment of existing neural structures to support new purposes.

The indeterminacy of computation might be an important ingredient in the re-deployment of existing structures. We advance the following speculative hypothesis (our aim is simply to articulate a fascinating hypothesis, not to advance empirical evidence for it).

The Massive Multiple Specifiability (MMS) Hypothesis. A multiply specifiable neural structure can be used to serve a range of different purposes (within the cognitive, perceptual, affective, and regulatory domains, for instance) by exploiting the different computations performed by the structure; and this may occur extensively throughout the brain.

Coupling the MMS hypothesis with the foregoing picture of different computations being tapped off a single neural structure gives the following possible sketch of how such plasticity might be achieved. The structure is situated in two or more distinct upstream–downstream milieux: each upstream–downstream milieu serves to tap off one of the computations that the structure is performing. How the upstream–downstream mechanisms are themselves controlled is, at this stage, a matter for pure speculation. An upstream–downstream mechanism might remain

quiescent until triggered by a specific event. Conceivably, the triggering event might toggle between the upstream–downstream mechanisms, switching one on and the others off. Alternatively, the upstream–downstream mechanisms might always be active, in which case the computations are tapped off concurrently, rather than sequentially. Via the postulated upstream–downstream mechanisms and their controlling mechanisms, the single neural structure is efficiently utilised to serve different biological purposes.

An investigation of the MMS hypothesis would form a fascinating new research programme in computational neuroscience. It probably would not be easy—Gabbiani's investigation of a *single* neurone has taken more than 20 years. Any results from the MMS hypothesis, positive or negative, are likely to be hard won. Nevertheless, multiple specifiability and the indeterminacy of computation provide an intriguing new theoretical angle on findings concerning neural plasticity.

6. Conclusion

The first part of this article provided a new conceptual framework, the multiple specifiability framework, characterising the phenomenon of the indeterminacy of computation—a phenomenon that is prevalent in Boolean systems but by no means limited to them.

We spelled out two major implications of indeterminacy for the study of cognition. First, there is the issue of the *extra explanatory step*. Researchers will need to take indeterminacy into account when framing computational explanations of both biological and artificial systems. Moreover, we suggested that hypothesising the existence of indeterminacy in a given neural structure may prove a valuable heuristic for investigating the role of structures peripheral to the given structure.

Second, there is the issue of neural *plasticity*. The brain may put the phenomenon of indeterminacy to good use. If the brain—as it reconfigures itself adaptively in response to growing processing demands and limited computational resources—pursues a strategy of

plasticity, then this may lead to numerous multiply-specifiable neuronal structures each serving more than one cognitive (or regulatory or affective) purpose, and doing so by means of simultaneously computing a number of different functions.

The strategy of achieving plasticity via indeterminacy may also prove useful in engineering artificial cognitive systems.

Acknowledgements

Many people have contributed to this paper through lively discussions and invaluable comments on earlier versions. We thank them, and several anonymous referees. This research was partly supported by the Israel Science Foundation Grant 386/20 to the first author, and a stipendiary fellowship from the Kreitman School of Advanced Graduate Studies at Ben-Gurion University of the Negev.

References

- Anderson, M. L. (2014). *After phrenology: neural reuse and the interactive brain*. The MIT Press.
- Badura, C., & Berto, F. (2019). Truth in Fiction, Impossible Worlds, and Belief Revision. *Australasian Journal of Philosophy*, *97*(1), 178–193.
- Bishop, J. M. (2009). A cognitive computation fallacy? Cognition, computations and panpsychism. *Cognitive Computation*, *1*(3), 221–233.
- Block, N. (1990). Can the Mind Change the World? In G. Boolos (Ed.), *Meaning and method: essays in honor of Hilary Putnam* (137–170). Cambridge University Press.
- Busemeyer, J. R., & Diederich, A. (2010). *Cognitive modeling*. Sage.
- Carandini, M., & Heeger, D. J. (1994). Summation and division by neurons in primate visual cortex. *Science*, *264*(5163), 1333–1336.
- Carandini, M., & Heeger, D. J. (2012). Normalization as a canonical neural computation. *Nature Reviews Neuroscience*, *13*(1), 51–62.

- Chalmers, D. J. (1996). Does a rock implement every finite-state automaton? *Synthese*, *108*(3), 309–333.
- Coelho Mollo, D. (2017). Functional individuation, mechanistic implementation: the proper way of seeing the mechanistic view of concrete computation. *Synthese*. 195(8), 3477-3497.
- Copeland, B. J. (in progress). Computational Levels and the Indeterminacy of Computation.
- Crama, Y., & Hammer, P. L. (2011). *Boolean functions: theory, algorithms, and applications*. Cambridge University Press.
- Cvitanović, T., Reichert, M. C., Moškon, M., Mraz, M., Lammert, F., & Rozman, D. (2017). Large-scale computational models of liver metabolism: how far from the clinics?: Cvitanović et al. *Hepatology*, 66(4), 1323–1334.
- Dennett, D. C. (1978). *Brainstorms: philosophical essays on mind and psychology*. The MIT Press.
- Dennett, D. C. (2013). *Intuition pumps and other tools for thinking*. W.W. Norton & Company.
- Dewhurst, J. (2018). Individuation without Representation. *The British Journal for the Philosophy of Science*, *69*(1), 103–116.
- Enroth-Cugell, C., & Robson, J. G. (1966). The contrast sensitivity of retinal ganglion cells of the cat. *The Journal of Physiology*, *187*(3), 517–552.
- Fresco, N. (2010). Explaining Computation Without Semantics: Keeping it Simple. *Minds and Machines*, *20*(2), 165–181.
- Fresco, N. (2014). *Physical computation and cognitive science* (Vol. 12). Springer.

- Fresco, N. (2015). Objective Computation Versus Subjective Computation. *Erkenntnis*, *80*(5), 1031–1053.
- Fresco, N. (forthcoming). How Context can Determine the Identity of Physical Computation. In M. Hemmo, S. Ioannidis, O. Shenker, and G. Vishne (Eds.), *Levels of Reality in Science and Philosophy*.
- Fresco, N, Wolf, M. J, & Copeland, B. J. 2016. On The Indeterminacy of Computation. The Annual Meeting of the International Association for Computing and Philosophy, University of Ferrara, Italy.
- Frigg, R. (2010). Models and fiction. *Synthese*, *172*(2), 251–268.
- Gabbiani, F., Krapp, H. G., Koch, C., & Laurent, G. (2002). Multiplicative computation in a visual neuron sensitive to looming. *Nature*, *420*(6913), 320–324.
- Gabbiani, F., Krapp, H. G., & Laurent, G. (1999). Computation of Object Approach by a Wide-Field, Motion-Sensitive Neuron. *Journal of Neuroscience*, *19*(3), 1122–1141.
- Hatsopoulos, N., Gabbiani, F., & Laurent, G. (1995). Elementary Computation of Object Approach by a Wide-Field Visual Neuron. *Science*, *270*(5238), 1000–1003.
- Jones, P. W., & Gabbiani, F. (2012). Logarithmic Compression of Sensory Signals within the Dendritic Tree of a Collision-Sensitive Neuron. *Journal of Neuroscience*, *32*(14), 4923–4934.
- Knill, D. C., and Pouget, A. (2004). The Bayesian Brain: The Role of Uncertainty in Neural Coding and Computation. *Trends in Neurosciences*, 27(12), 712–719.
- Koch, C. (1999). *Biophysics of computation: information processing in single neurons*. Oxford University Press.
- Lewis, D. (1978). Truth in Fiction. *American Philosophical Quarterly*, *15*(1), 37–46.

- Lewis, D. (1983). Postscripts to "Truth in Fiction." In *Philosophical Papers Volume I* (pp. 276–280). Oxford University Press.
- McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*(4), 115–133.
- Menon, V., & Uddin, L. Q. (2010). Saliency, switching, attention and control: a network model of insula function. *Brain Structure and Function*, *214*(5–6), 655–667.
- Papayannopoulos, P., Fresco, N. & Shagrir, O. (in progress). On Two Different Kinds of Computational Indeterminacy.
- Piccinini, G. (2015). *Physical Computation: A Mechanistic Account*. Oxford University Press.
- Piccinini, G., & Bahar, S. (2013). Neural Computation and the Computational Theory of Cognition. *Cognitive Science*, *37*(3), 453–488.
- Price, C. J., & Friston, K. J. (2005). Functional ontologies for cognition: The systematic definition of structure and function. *Cognitive Neuropsychology*, *22*(3–4), 262–275.
- Proudfoot, D. (2006). Possible worlds semantics and fiction. *Journal of Philosophical Logic*, *35*(1), 9-40.
- Putnam, H. (1988). *Representation and reality*. The MIT Press.
- Shagrir, O. (2001). Content, computation and externalism. *Mind*, *110*(438), 369–400.
- Shagrir, O. (2020). In defense of the semantic view of computation. *Synthese*, *197*(9), 4083–4108.
- Sorensen, R. A. (1999). Mirror Notation: Symbol Manipulation without Inscription Manipulation. *Journal of Philosophical Logic*, *28*(2), 141–164.
- Sprevak, M. (2010). Computation, individuation, and the received view on representation. *Studies in History and Philosophy of Science Part A*, *41*(3), 260–270.

- Turing, A. M. (1948). Intelligent Machinery. In B. J. Copeland (Ed.), *(2004) The Essential Turing* (pp. 410–432). Oxford University Press.