CONNECTIONISM, ANALOGICITY AND MENTAL CONTENT

GERARD O'BRIEN

Department of Philosophy
The University of Adelaide
South Australia 5005
AUSTRALIA

Acta Analytica Vol. 22 (1999): pp.111-31

CONNECTIONISM, ANALOGICITY AND MENTAL CONTENT

Abstract

In *Connectionism and the Philosophy of Psychology*, Horgan and Tienson (1996) argue that cognitive processes, pace classicism, are not governed by exceptionless, "representation-level" rules; they are instead the work of defeasible cognitive tendencies subserved by the non-linear dynamics of the brain's neural networks. Many theorists are sympathetic with the dynamical characterisation of connectionism and the general (re)conception of cognition that it affords. But in all the excitement surrounding the connectionist revolution in cognitive science, it has largely gone unnoticed that connectionism adds to the traditional focus on computational *processes*, a new focus – one on the *vehicles* of mental representation, on the entities that carry content through the mind. Indeed, if Horgan and Tienson's dynamical characterisation of connectionism is on the right track, then so intimate is the relationship between computational processes and representational vehicles, that connectionist cognitive science is committed to a *resemblance* theory of mental content.

1 Introduction

In *Connectionism and the Philosophy of Psychology*, Terrence Horgan and John Tienson (henceforth H&T) set out to develop a comprehensive characterisation of connectionism that both marks it as a rival to its classical predecessor and explains why it is superior (1996). The result is a framework they call *dynamical cognition*, a conception of mind at whose heart is the rejection of the discrete mathematics of digital computation in favour of the fundamentally continuous mathematics of dynamical systems theory. Cognitive processes, pace classicism, are not governed by precise, exceptionless, "representation-level" rules; they are instead the work of defeasible cognitive tendencies subserved by the non-linear dynamics of the brain's neural networks.

Along with many theorists, I am sympathetic with the dynamical characterisation of connectionism and the general (re)conception of cognition that it affords. What is more, I am especially sympathetic with H&T's focus on "content-appropriate" cognitive forces, physically realised as complex patterns of network activation and connectivity, competing and combining with one another to determine the temporal evolution of cognitive behaviour (1996, chp.4). This, I believe, is the proper treatment of connectionism. But revolutions often have unforeseen consequences – consequences that even those in the vanguard fail to appreciate. And in the case of the connectionist revolution in cognitive science, one major consequence has largely gone unnoticed. This is that connectionist cognitive science is committed to a distinctly old-fashioned and, for much of this century at least, unpopular theory of mental representation. In a nutshell, if H&T's dynamical characterisation of connectionism is on the right track, then connectionists must embrace a *resemblance* theory of mental content.

This is a conclusion for which most philosophers of cognitive science are quite unprepared. As H&T observe, it has been a working assumption in this field that the issue of intentionality and its naturalisation are orthogonal to debates about cognitive architecture:

> [T]hroughout this book we have taken representation for granted. In doing so, we simply follow connectionist (and classicist) practice…. It is assumed that natural cognitive systems have intentional states with content that is not derived from the interpreting activity of other intentional agents. But it is not the business of either classical cognitive science or connectionist cognitive science to say where underived intentionality "comes from" (although each may place certain constraints on an answer to this question). (1996, p.13)

But it will be the argumentative burden of this paper to show that, along with the many things that change with connectionism, this assumption no longer holds good. Connectionism adds to the traditional focus in cognitive science on the *processes* by which mental representations are computationally manipulated, a new focus – one on the *vehicles* of mental representation, on the entities that carry content through the mind. Indeed, the relationship between computational processes and representational vehicles is so intimate within connectionist cognitive science, that the story about the former implies one about the latter. As a consequence, while classicism is compatible with a range of different theories of mental content, connectionism, in virtue of its dynamical style of computation, doesn't have this luxury.

To see why all of this is so, however, we will need to go back to the beginning. And in the beginning there was computation.

2. What is Computation?

Jerry Fodor is fond of remarking that there is only one important idea about how the mind works that anybody has ever had. This idea he attributes to Alan Turing:

> [G]iven the methodological commitment to materialism, the question arises, how a machine could be rational?…Forty years or so ago, the great logician Alan Turing proposed an answer to this question…Turing noticed that it isn't strictly true that states of mind are the only semantically evaluable material things. The other kind of material thing that is semantically evaluable is *symbols*…. Having noticed this parallelism between thoughts and symbols, Turing went on to have the following perfectly stunning idea. "I'll bet", Turing (more or less) said, "that one could build a *symbol manipulating machine* whose changes of state are driven by the material properties of the symbols on which they operate (for example, by their weight, or their shape, or their electrical conductivity). And I'll bet one could so arrange things that these state changes are rational in the sense that, given a true symbol to play with, the machine will reliably covert it into other symbols that are also true. (Fodor, 1992, p.6)

The rest, as one says, is history. Turing's idea very soon led to the development of digital computers, and the theory of digital computation when applied to the mind leads to classicism. All of this is old hat. But in the subsequent discussion of Turing's insight and its implications for cognitive science, one detail is sometimes overlooked. This is that there are actually two ideas embodied in this aspect of Turing's work; two ideas that can and should be distinguished. One is a general idea about *computation*; the other is a more specific idea about how to construct a *computational machine.*

In 1936, when the perfectly stunning idea alluded to by Fodor was taking shape in Turing's mind, the word 'computer' meant nothing more than "a person doing mathematical calculations" (see Hodges, 1983, chp.2). What Turing did, as everyone knows, was to try to imagine a

computational machine: a machine that could perform such calculations automatically, without human intervention. The first step in Turing's imagining, however, was to consider not *how* such a machine could be constructed, but *what* such a machine had to do. Turing asked: what, from an objective point of view, occurs when *we* perform a mathematical calculation? His answer was: we write a sequence of symbols on a piece of paper. But not just any old sequence of symbols. Our behaviour constitutes "computing" only when the later symbols in the sequence are systematically related to the earlier ones, the precise relation depending on the kind of mathematical operation being performed. For example, in the case of a simple arithmetical calculation, such as adding two numbers together, the third symbol we write refers to a number that is the *sum* of the two numbers denoted by the first and second symbols we have written. Seen from this perspective, a computation is a process that produces a *mathematically coherent sequence of symbols*, where this is a sequence in which later symbols bear comprehensible mathematical relations to earlier ones. This is the general idea about computation that is embodied in Turing's work.

That computational processes are mathematically coherent sequences of symbols is not an idea that originated with Turing, of course. This is just Turing's analysis of calculational procedures with which we are all familiar. The idea that did originate with Turing is the more specific one about how to construct a machine that performs such computations, that automatically generates such a sequence of symbols. That is, Turing's great achievement wasn't that he told us what computational processes *are*; his achievement was to show us one way that these processes could be *mechanised* (more about which in the next section).

Distinguishing between these two ideas is important, for the following reason. Once it is clear that the conception of computation that Turing employed in his work is quite distinct from his more specific idea as to how computational processes could be mechanised, it is possible to investigate the former independently of the latter (something that is sorely lacking in most contemporary discussions in this area). When this is done, it quickly becomes apparent that even this general conception of computation is too restrictive, and can be liberalised along two dimensions: one concerning the entities over which computational processes are defined; the other the systematic relations that obtain between these entities.

First, there seems to be no reason, even in the context of mathematical calculation, to restrict computation to sequences of *symbols*. Geometric proofs, for example, use a combination of symbols and nonsymbolic diagrams. And when we step outside the field of mathematics to consider the vast range of calculations we routinely perform in other domains (remembering all the while that it was to our calculational capacities that Turing looked for the conception of computation he employed), it is obvious that our deliberations engage all manner of nonsymbolic representational entities (such as when we draw a picture to determine whether A is taller than C when you are told that B is taller than C but shorter than A),. What seems essential to computation, therefore, is not that it implicates symbols as such, but that *representational vehicles* of some kind are employed, be these vehicles symbolic or nonsymbolic.

Second, there seems to be no reason to restrict the systematic relations between these representational vehicles to those characterisable in mathematical terms. The computational process at the centre of Fodor's discussion of Turing, for example, is deductive inference, where the systematic relation that obtains between later symbols (the conclusion) and earlier ones (the premises) is truth preservation. And there are other kinds of representational relations that are exploited in computation. Consider, as another example, the calculation we perform when using a street map to determine the best route across a city. One of the crucial things in this case, as we work out a route by tracing our finger across the map, is that the relations between the map's

representational vehicles (the black lines that represent streets, the red dots that represent traffic lights, and so forth) accurately reflect the relations between the map's representational objects (the streets, the traffic lights, and so on). In the end, perhaps the best we can say is that the representational vehicles participating in a computational process must be *semantically coherent*, in the sense that they bear comprehensible (ie, non-arbitrary) semantic relations to one another.

With these emendations in place, we arrive at a very general conception of computation as a procedure in which *representational vehicles are processed in a semantically coherent fashion*.[1] When this conception is applied to cognition, we arrive, in turn, at the *generic* version of the computational theory of mind: cognitive processes are semantically coherent operations over neurally implemented representational vehicles. This is a very good idea; it is the idea, moreover, which forms the foundation of cognitive science. But it is an idea that needs to be fleshed out; we need to know how computational processes, so understood, can be physically realised. This is where Turing does become important. For Turing showed us a way to bring this idea to life.

3. Turing's Way: Digital Computation and Classical Cognitive Science

A computational device is one that mechanistically processes representational vehicles in a semantically coherent fashion. But how can such a device be constructed? How, to put this another way, can the semantic properties of physically realised representational states shape the mechanistic processes by which they are processed? The short answer here is that they can't – semantic engines are impossible. The trick to mechanising computation, therefore, is to align the physical properties of a physical device with semantic ones, such while its causal behaviour is entirely determined by the former, it nonetheless respects the latter. The trick to mechanising computation, in other words, is to make a physical engine behave *as if* it were a semantic one.

Turing saw a way of doing this. What is needed, he reasoned, is a systematic means of physically encoding information, and a mechanism for manipulating this representational medium in a semantically coherent fashion. The first requirement, he thought, could be satisfied by written symbols, and the second by a mechanism that recognises and transforms these symbols purely on the basis of their material properties, but in accordance with rules that ensure that these mechanical symbol manipulations are semantically coherent. Turing's solution was the specification of an abstract machine – the conceptual basis of the *digital* computer – replete with a tape on which symbols are written, and a read/write head so configured that it behaves as if it were following a set of primitive computational instructions.

Physically implementing Turing's abstract machine is no easy feat, however. One must first find a means of physically realising a symbolic representational medium. This is achieved by systematically partitioning some continuously variable physical property, and then providing these partitions and their concatenations with a semantic interpretation (whether this interpretation concerns numbers, propositions, or whatever). The partitioning of a continuously variable physical property generates *syntactic* structure, which under interpretation becomes a symbolic medium. In conventional digital computers this role is performed by the electrical voltages[2] between pairs of

[1] This more general conception of computation can be found, if one looks carefully enough, in a number of places in the philosophical literature. See, eg, Cummins and Schwarz, 1991, p.64; Dietrich, 1989; Fodor, 1975, p.27; and Von Eckardt, 1993, pp.97-116.

[2] Strictly speaking, electrical voltage is a continuously variable *physical magnitude*, where a physical magnitude is a function that systematically assigns numbers to a physical property or properties of a physical system. But certain computers use continuously variable physical properties to physically implement their representational media that are

wires (eg, strips of aluminium printed on silicon chips), which receive a binary partitioning into "high" and "low" states. Thus the symbols in such computers are physically inscribed in the form of concatenations of high and low voltages. One must next find a means of "reading" and "writing" the symbols in this representational medium. This requires a device capable of detecting and transforming these symbols on the basis of their syntactic properties, rather than their microphysical details. This is achieved in conventional digital computers by grouping transistors together to form "gates", which are differentially responsive to high and low voltage states, rather than precise voltage values.

Together these two steps lead to the construction of a *syntactic engine*: a physical device whose casual behaviour is determined by the syntactic properties of the symbols recorded in its representational medium. But left to its own devices, a syntactic engine is not terribly useful; it doesn't "naturally" transform symbols in a manner that is sensitive to their semantic content, and hence doesn't "naturally" perform any computations. In general, there is nothing intrinsic to the syntactic structure of symbols that ordains that they will be manipulated in a semantically coherent fashion (eg, there is nothing in the particular sequence of high and low voltages comprising one symbol in a conventional digital computer that dictates that it must bear some sensible semantic relation to the sequence of high and low voltages comprising another). Consequently, it takes work to harness the power of a syntactic engine; the engine must be forced to perform certain kinds of symbol manipulations (ie, those that are semantically coherent) and avoid others (those that aren't). The engine's behaviour must, in short, be *rule-governed*. These rules, while they are syntactically applied, are shaped in accordance with the semantic relations that obtain between the symbols on which the syntactic engine operates, and hence contain the semantic coherence of the device. They dictate that this engine will transform one symbol into another only if their is a sensible semantic relation between them. It is only when syntactic engines conform to such rules that they become digital *computers*.

In practice, of course, these rules, like every other feature of the device, must be physically implemented. It turns out that many of these rules can be explicitly coded in the form of symbols inscribed in a digital computer's representational medium (ie, the notion of a stored program). Ultimately, though, a set of primitive rules (analogous to the primitive computational instructions resident in the read/write head of Turing's abstract machine) must be "hardwired". This is done, in standard digital computers, by subtly wiring together groups of transistors – the read/write "gates" mentioned earlier – so that the voltage transformations they execute are always in accordance with the primitive rules. This microcircuitry thus literally embodies a set of primitive instructions and with it a basic computational competence

This is Turing's way of mechanising computation. And when Turing's way is applied to the mind, the result is classicism: the theory that cognitive processes are digital computations. Classicism takes the generic computational theory of mind (the doctrine that cognitive processes are semantically coherent operations over neurally implemented representational vehicles) and adds a more precise account of both the representational vehicles (they are complex symbol structures possessing a combinatorial syntax and semantics) and the computational processes (they are rule-governed, syntactical transformations of these symbol structures). The rich diversity of human

---

not easily characterisable in these terms. That is, the *variability* of a particular physical property is not always comfortably captured in a straightforwardly numerical fashion. To achieve the necessary generality, therefore, I will talk in terms of the physical properties that realise the representational media of computational systems, rather than in terms of physical magnitudes that numerically characterise these properties.

thought, according to classicism, is the result of a colossal number of syntactically-driven operations defined over complex neural symbols.

All of this is standard fare, as classicism is normally understood in the literature as the digital computational conception of mind. According to H&T, for example, the most fundamental assumption of classical cognitive science is a commitment to what they term *programmable representation-level rules*: "cognitive processing conforms to precise, exceptionless rules, statable over the representations themselves and articulable in the format of a computer program" (1996, p.24). Their focus on such rules is quite justified, given the foregoing description of digital computation, as it is these rules, *and these rules only*, that embody the computational competence of a digital device and ensure that it behaves in a semantically coherent fashion. Without these rules, a syntactic engine is utterly blind; with them, it behaves as if it were a semantic engine, and in so doing is capable of computational work.

So far, then, so good. However, what is not so clear, reading from the literature at least, is what happens if we decide that Turing's way is not the mind's way; that cognition is not, or not predominantly, rule-governed symbol manipulation. It is here that the failure to distinguish between the two ideas embodied in Turing's work is responsible for a good deal of confusion. For when some theorists in cognitive science come to reject classicism, as many have been doing recently, there is a tendency for them to reject too much. When they ask "What might cognition be, if not [classical] computation?" (Van Gelder, 1995), their answer tends towards a radical, non-representationalist form of dynamical systems theory which discards the whole idea of semantically coherent processing, and with it the only idea we have about how intelligent behaviour could arise in the physical world. Thus it's important to realise that the theoretical landscape is more extensive than their thinking implies; that one can reject classicism without rejecting a computational conception of cognition.

4. An Alternative Way: Analog Computation and Connectionism

Everyone knows that computer science routinely distinguishes between digital and analog computers. So at first glance it would seem that, in analog computation, there is indeed an alternative means of mechanising computation. But what are analog computers and how do they differ from digital devices? The standard account of the distinction between digital and analog computers in the literature centres on the representational vehicles a computational device deploys. If the physical material that implements each of these vehicles can vary to some extent without thereby affecting its representational content, the device is said to employ a *discrete* representational medium, and hence is digital. If, on the other hand, any variation in this physical material constitutes a variation in representational content, the medium is *continuous* and the device is thought to be analog.[3]

As it stands, however, this way of drawing the distinction is not unproblematic, largely because it is sometimes unclear whether a particular representational medium is more discrete or continuous. The legitimacy of the distinction has thus been called into question, with considered

[3] One encounters different versions of this distinction digital and analog computers in the literature. For example, some computer scientists talk in terms of the difference between modelling a physical process with real numbers (analog computation) and with rational number approximations (digital computation). Others invoke a distinction between discrete symbol manipulation (digital) and the manipulation of real-valued signals (analog). The version in the text is closer in spirit to the way this distinction is developed in philosophy, mainly by Nelson Goodman (1969, chp.4) and David Lewis (1971).

opinion varying from those who think the distinction is merely relative to our explanatory purposes and interests, to those who argue that the (ultimately discrete) nature of our world dictates that the class of analog computers is empty. Furthermore, even if this way of drawing the distinction can be made to work, it isn't terribly illuminating. For what we really want to know, and what the standard story doesn't tell us, is how the *causal operation* of analog computers differs from their digital counterparts.

Fortunately, there is such an account of the distinction between digital and analog computers in the offing; one, moreover, that because it focuses on causal operation is able to explain why digital devices deploy discrete representational media while analog machines tend towards more continuous forms of information coding. The key here, not surprisingly, is semantically coherent behaviour and how it is achieved. We've already seen how digital computers do it: they have semantic coherence imposed on them by a set of syntactically applied, symbol transformation rules. Analog devices, as we are about to see, do it differently.

Consider a standard example of an analog computer: a scale model of a proposed building, onto which light is shone in order to determine shadow patterns. In what sense is this device mechanistically processing representations in a semantically coherent fashion? First, a number of its component parts are representational: the light shone on the model represents sunlight, the model building represents a planned real building, and the shadow patterns generated represent the shadow patterns that would be produced by sunlight shining on the actual building. Secondly, these representational vehicles enter into causal relations with one another: the light shone on the model building causes specific kinds of shadow patterns to be produced. Finally, and crucially, the causal relations that obtain between these representational vehicles are semantically coherent: the content of the output representation (the shadow pattern) is non-arbitrarily related to the contents of the input representations (the light and the model building), in that sunlight falling at that angle on a real building of that shape would produce that shadow pattern.

But from whence does this semantic coherence come? The operating principles in the case of a scale model are obvious and straightforward. The model derives its computational power from the *similarities* that exist between the representational elements of its material substrate and the representational domain its computations are about. Specifically, the light shone on the scale model behaves the *same* way as sunlight, and the shape of the model building is the *same* as that of the proposed building. These similarities ensure that the physical relations and hence causal dynamics of the "input" representational vehicles in the scale model (the light and the model building) *automatically track* those between the objects in the represented domain (sunlight and the proposed building), such that the "output" representational vehicle of this analog device (the shadow pattern across the model) mirrors the behaviour of its representational object in this domain (the shadow pattern that would occur in the real world).

While a scale model is a very simple analog computer, its operating principles exemplify the basis of all analog computation. Baldly stated, analog computers compute by physically manipulating "analogs" of their representational domains. A material substrate embodies an analog of some domain when there is a *structural isomorphism* between them, such that elements of the former (the representational vehicles) *resemble* aspects of the latter (the representational objects).[4]

[4] It's important to stress the relevant isomorphism here is between representational vehicles and their objects, rather than between the *causal network* of vehicles and *certain relations* (causal or otherwise) between their representational objects. The latter is sometimes termed *functional isomorphism* (see, eg, Von Eckhardt, pp.206-14), and is another way of construing semantically coherent processing. In all computations there is a functional isomorphism between representational vehicles and their objects. The question, though, is how this functional isomorphism is generated. In

The resemblance relation here can come in different varieties. In the case of the scale model, the relation is one of *first order* structural isomorphism, whereby the relevant physical properties of the representational objects (sunlight, shape of the proposed building, etc) are represented by *equivalent* physical properties of the representational vehicles (the light source, the shape of the model building, etc.). But this particular requirement can be relaxed without undermining resemblance. For example, we might suppose that certain properties of representational objects can be represented by non-equivalent properties of representational vehicles, *as long as variations in the former are systematically mirrored by corresponding variations in the latter*. This gives us a notion of *second order* structural isomorphism (see Palmer, 1978; and Shepard and Chipman, 1970). Whether the resemblance relation is first or second order, however, its obtaining makes it possible for the material substrate to behave in a semantically coherent fashion. That is, because its representational vehicles resemble their representational objects, the causal relations between the former can naturally reflect certain relations (causal or otherwise) between the latter (see Trenholme, 1994).

The computational operations of an analog computer are driven by the natural causal laws that apply to (aspects of)[5] its material substrate. This substrate must possess sufficiently complexity and variability to be capable of representing all of the relevant properties of the target domain over which one wants to compute, together with their variation. This demands an intimate relationship between the computer's substrate and the representational vehicles it implements, an intimacy that in practice results in a more *continuous* representational medium (such that fine-grained variations in the substrate are representationally and hence computationally significant). All this makes analog computation very different from digital computation. In the latter case, a material substrate undergoes semantically coherent behaviour, not in virtue of a structural isomorphism between its representational vehicles and its representational domain, but by being forced to conform to a set of computational rules. Because these computational rules are quite distinct from the causal laws that govern the behaviour of the (continuously variable) physical properties comprising the material substrate, their enforcement is achieved, as we have seen, through a syntactic partitioning of this substrate. It is the satisfaction of this requirement that gives rise, in digital computers, to a more *discrete* representational medium.

Analog computation is the dynamical alternative to Turing's way. An analog conception of cognition would thus represent an alternative to classicism. But what would such an alternative look like? One answer, or at least I shall now argue, is connectionism.[6]

Whereas classicism is grounded in the computational theory underpinning the operation of digital computers, connectionism relies on a neurally inspired computational framework commonly

---

digital computation the isomorphism is imposed by syntactically applied rules. With analog computation, by contrast, functional isomorphism is founded on structural isomorphism.

[5] Not *all* of the physical properties of an analog computer's material substrate are relevant insofar as its computational operations are concerned. It doesn't matter, for example, from what material a scale model of a proposed building is constructed, as long as the material is opaque. This is because the physical property that implements the relevant representational vehicle here is something like *opaque shape* (for want of a better term), and many different materials, possessing all manner of different physical properties, can exhibit *this* physical property.

[6] Connectionism represents *one* analog conception of mind, not *the* analog conception of mind (in contrast with the way that classicism is *the* digital conception of mind). This is because connectionism is the analog theory of mind one gets when one assumes that the *neural network* level of description of the brain is the right level for understanding cognition. Other analog conceptions of mind, focusing on different levels of description, are conceivable (though highly implausible, in my view).

known as *parallel distributed processing* (or just PDP).[7] A PDP network consists in a collection of processing units, each of which has a variable activation level. These units are physically linked by connections, which enable the activation level of one unit to contribute to the input and subsequent activation of other units. These connections incorporate modifiable connection weights, which modulate the effect of one unit on another in either an excitatory or inhibitory fashion. A PDP network typically performs computational operations by "relaxing" into a stable pattern of activation in response to a stable array of inputs. Human cognitive processes, according to connectionism, are the computational operations of a multitude of PDP networks implemented in the neural hardware in our heads. And the human mind is viewed as a coalition of interconnected, special-purpose, PDP devices whose combined activity is responsible for the rich diversity of our thought and behaviour. This is the connectionist computational theory of mind.

A strong hint that connectionism represents an analog conception of mind can be found in the intimate relation between the PDP computational framework and the neuroanatomy of the brain (something that is utterly lacking in digital models of cognition). Rumelhart and McClelland, for example, have been explicit about the fact that PDP systems directly model certain high-level physical properties of real neural networks. Most obviously, the variable activation levels of processing units and the modifiable weights on connections in PDP networks directly reflect the spiking frequencies of neurons and the modulatory effects of synaptic connections, respectively.[8] Terrence Sejnowski goes further, arguing that while PDP systems do not attempt to capture molecular and cellular detail, they are nonetheless "stripped-down versions of real neural networks similar to models in physics such as models of ferromagnetism that replace iron with a lattice of spins interacting with their nearest neighbors" (1986, p.388). As with any idealisation in science, what goes into such an account depends on what properties of neural nets one is trying to capture. The idealisation must be complex enough to do justice to these properties, and yet simple enough that these properties are sufficiently salient (see, eg, Churchland & Sejnowski 1992, chp.3). In this respect, the PDP framework isolates and hence enables us to focus on the computationally significant properties of neural nets, while ignoring their fine-grained neurochemistry. Our best neuroscience informs us that neural nets compute by generating patterns of neural activity in response to inputs, and that these patterns of activity are the result of the modulatory effects of synapses in the short-term, and modifications to these synapses over the longer term. It's precisely these structural and temporal properties that are captured by the networks of processing units and connection weights that comprise PDP systems.

In spite of this strong hint, however, most theorists have been reluctant to press the concept of analogicity into service in their descriptions of how connectionism differs from classicism. This is due primarily, I think, to the fact that the PDP computational framework resists easy classification on the standard account of the digital/analog distinction that one encounters in the literature: since the material substrate of a PDP model comprises connection weights and unit activation values that need not take on continuous values, it's often not obvious whether this substrate realises a continuous or discrete representational medium. But this all changes when we move to the alternative conception of the digital/analog distinction that I've outlined. Viewed from this

[7] The locus classicus of PDP is the two volume set by Rumelhart, McClelland, and the PDP Research Group (Rumelhart & McClelland, 1986; McClelland and Rumelhart, 1986).

[8] See, eg, Rumelhart and McClelland 1986, chp.4. There are other connectionists who are not entirely happy with this interpretation, however. Paul Smolensky, for example, argues that because we are still largely ignorant about the dynamical properties of the brain that drive cognitive operations, and because the PDP framework leaves out a number of properties of the cerebral cortex, connectionism is situated at a level once removed from real neural networks (1988).

perspective, the issue is not whether these models employ continuous or discrete representational media, but how they produce semantically coherent behaviour. And the PDP models discussed in the connectionist literature do this by first constructing and then exploiting structural isomorphisms between their substrates and their representational domains, not by performing rule-governed manipulations of symbolic representations. It is clear, in other words, that PDP networks, at least as they are standardly employed, [9] are analog computational devices.[10]

Consider, as a simple example, the mine/rock detecting network discussed by Paul Churchland (1988, pp.157-62). The network takes as input a "sonar echo" emanating from either a rock or a mine (sampled across 13 different frequencies and thus coded across 13 input units), processes this through a hidden layer (of 7 hidden units), and produces an output (across 2 output units) which should indicate the source of the echo. The network is initially exposed to a range of rock echoes (which are superficially different) and a range of mine echoes (likewise), and is trained up using the back propagation learning procedure. This procedure so modifies the network's connectivity pattern, that it soon becomes capable of distinguishing between the two general types of echoes. But how does it do this? Numerical analysis of the activation patterns generated over the hidden units of the trained up network reveals that they cluster in two disjoint regions of activation space, and depending on which region is activated the network outputs either "rock" or "mine". The back propagation learning procedure has thus moulded the network's material substrate, through subtle modifications to its pattern of connectivity, that it now contains an activation landscape that systematically mirrors abstract properties of the "sonar echoscape" (for want of a better term). In other words, the trained up network embodies an analog of its representational domain, in that its representational vehicles (the connection weight representations stored in its pattern of connectivity, which manifest themselves in the form of network activation patterns) are structurally isomorphic (in a second order sense) with objects in this target domain (the "abstract" properties in common to mine and rock echoes, respectively).

Or consider, as another example, NETtalk, probably the most talked about PDP model in the connectionist literature (Sejnowski and Rosenberg, 1987). NETtalk transforms English graphemes into their appropriate phonemes, given the context of the words in which they appear. The task domain, in this case, is even more abstract, in that it is the letter-to-sound correspondences that exist in the English language. But NETtalk's operating principles are similar to those found in the

[9] This qualification is necessary, of course, because it is well known that a PDP network can implement a Turing machine. When employed in this fashion, PDP networks operate as digital computers.

[10] The analog character of the PDP framework is also obscured by the fact that most PDP networks are *simulated* on digital computers. In such simulations, the activation values that compose a network's activation pattern together with the values of the connection weights are typically recorded in complex arrays, each of whose elements is subject to updating according to the algorithms that model the network's activity. But these data structures are not equivalent to a pattern of activation or a the configuration of connection weights across a real (non-simulated) PDP network. The latter are structures constructed from physically connected elements (such as neurons), each of which realises a continuously variable physical property (such as a spiking frequency) of a certain magnitude. The former, by contrast, are *symbolic representations* of such structures, in that they consist of a set of discrete symbol structures that "describe" in a numerical form the individual activation levels of a network's constituent processing units together with the values of its connection weights. The activation landscape embodied in a real network thus has a range of complex structural properties (and consequent causal powers) that are not reproduced by the data structures employed in simulations. This fact is most vividly demonstrated by the temporal asymmetries that exist between real PDP networks and their digital simulations: the simulations are notoriously slow at processing information, when compared to their real counterparts, in spite of the incredible computational speed of the digital machines on which they are run. The bottom line here is that a simulated network does not embody an analog of its representational domain, and hence is no more an analog device than a simulated hurricane is a hurricane.

rock/mine detector. Again, back propagation is used to shape its activation landscape, this time consisting of patterns across 80 hidden units, such that, eventually, a structural isomorphism obtains between its representational substrate and the target domain. It is this isomorphism that is revealed in the now very familiar cluster analysis to which Sejnowski and Rosenberg subjected NETtalk. It is this isomorphism that makes it right and proper to talk, as everyone does, of a *semantic metric* across NETtalk's activation landscape. Furthermore, it is this isomorphism that provides NETtalk with its computational power: when the light of a grapheme, embedded in an array of graphemes, is shone on NETtalk's surface, it automatically casts the appropriately contextualised phonemic pattern. This PDP network is thus an analog computational device in just the same way as the scale model we examined earlier: its material substrate embodies a complex "scale model" of the letter-to-sound correspondences found in English.

Nothing could be clearer than that these networks do not transform symbols according to syntactically applied rules; than that they are not digital computers. Yet they process their representational vehicles in a semantically coherent fashion. They are able to do this because their material substrates have been moulded by learning procedures into "shapes" that resemble aspects of their representational domains. When such resemblance relations are in place, their is no need to force these networks to conform to rules which dictate how their representational vehicles are to be processed. Instead, semantic coherence emerges quite naturally in these networks, driven by the causal laws that apply to the materials from which they are constructed.

In summary, the standard PDP networks discussed in the literature are analog computational devices. And if PDP networks are analog devices, then connectionism, the theory one gets when one applies the PDP computational framework to cognition, is an analog conception of mind.

5. The Cost of Connectionism: A Resemblance Theory of Mental Content

According to H&T, what sets connectionism apart from classicism is its eschewal of the discrete mathematics of digital computation in favour of the fundamentally continuous mathematics of dynamical systems theory; cognitive processes are not governed by exceptionless, "representation-level" rules, they are the work of defeasible cognitive tendencies subserved by the non-linear dynamics of the brain's neural networks. (1996, Chps.2&4). To this extent, their dynamical characterisation is quite consistent with the analog reading of connectionism developed in the previous section. But dynamical systems theory, on its own, is not a computational framework. There is no fundamental difference, from a purely dynamical perspective, between NETtalk and the convection currents generated in a pot of boiling water. In this sense, dynamical systems theory dissolves the distinction between intelligent and unintelligent behaviour, and hence is quite incapable, without supplementation, of explaining cognition. In order for dynamical engines to be capable of driving intelligent behaviour they must do some computational work: they must learn to behave as if they were semantic engines.

H&T recognise this fact. They recognise that as classicists have a robust story to tell about how semantic coherence is achieved in cognition (or "content-appropriateness" to use their term), connectionists must do the same. The story they subsequently tell (see especially 1996, Chps.6&9) focuses on the role of "content-appropriate cognitive forces", subserved by spreading activation across PDP networks. "Certain kinds of content-relevant interaction are automatic for systems that have states that emit content-relevant cognitive forces", H&T claim, which is a key difference from classical systems "in which the operative PRL [ie, programmable representational level] rules must determine all such outcomes" (1996, p.99). But exactly how are such content-appropriate forces realised in PDP networks? The answer H&T provide is that in training up a network, the

representations that comprise cognitive states are placed *strategically*, rather than *arbitrarily*, on an activation landscape, such that "their relative-position relations systematically reflect key semantic properties and relations of the relevant individual cognitive states themselves" (1996, p.156).

This is where H&T leave it. But one is entitled to ask for more. In particular, one is entitled to ask how an activation landscape could be so arranged that its various regions reflect the semantic relations to which H&T refer. The only answer that would seem to be available here, the only answer that entitles connectionists to dispense with representation-level rules, is one that invokes a structural isomorphism between this activation landscape and its representational domain. It's this isomorphism that renders the shape of the activation landscape semantically significant, and hence endows the points in this landscape with the content-appropriate causal powers. Dynamical engines, left to their own resources, might be incapable of engaging in semantically coherent behaviour; but once these engines are coupled to an analog of some region of the world, they are transformed into powerful computational devices. H&T's dynamical characterisation of connectionism is thus based, tacitly if not explicitly, on analog computation.

The implications for connectionist cognitive science are many. Most striking of all, I think, is that connectionism brings with it a new focus on the representational *vehicles* implicated in cognition, in addition to the conventional focus on the computational *processes* in which they are involved. This is because the semantic coherence of cognition, according to connectionism, is actually embodied in these vehicles, rather than in a set of independently defined computational rules (independent, that is, of the representational substrate of cognition). To put it bluntly, the "shape" of these vehicles matters; it's their "shape" that drives cognition.[11] And this completely changes our view of one of the most important topics in the contemporary philosophy of mind: the determination of mental content.

Philosophers, as H&T observe (1996, p.13), have long thought that the issue of intentionality is orthogonal to the question of cognitive architecture. This view is the legacy of classical cognitive science. Given that digital computations inherit their semantic coherence from rules that are quite distinct from the structural properties of the symbols they apply to, classicism appears to place few constraints on a theory of mental content. As long as these symbols are transformed according to these rules, it matters not where their representational content derives from. Thus the causal, functional and teleofunctional theories of content that dominate the current literature are all, prima facie, compatible with the idea that mental representations are symbols. In fact, of all the psychosemantic theories in the philosophical marketplace, there is only one that would seem incompatible with classicism. This is the resemblance theory of mental content, according to which representational vehicles are contentful in virtue of resembling, in some way, their representational objects. As Robert Cummins notes, "[classical] computationalists must dismiss similarity theories of representation out of hand; nothing is more obvious than that data structures don't resemble what they represent" (1989, pp.30-1).

But all of this changes when we replace classicism with connectionism. Connectionism, because it is grounded in analog computation, and because analog computation requires the presence of a structural isomorphism between representational substrate and target domain, completely closes the gap between the issues of intentionality and cognitive architecture. In

[11] This is not to say that the "shape" of mental symbols is wholly unimportant in the classical theory of mind. But the importance of shape in this latter case is relative – relative to the rules according to which these symbols are transformed. The point is that so long as these symbols are transformed according to these rules, it doesn't matter what shape they are.

connectionism, the content a mental representational vehicle carries through the mind cannot be independent of its "shape". As a consequence, connectionists just don't have the same luxury as classicists when it comes to mental content determination. They are forced to explain mental content, at least in part, in terms of resemblance relations between representational vehicles and their representational objects. They are forced, in short, to embrace a resemblance theory of mental content.

A few years ago this would have seemed a serious objection to connectionism. The resemblance theory was thought to suffer from a number of fatal flaws, and hence, in most philosophers' minds, was not much more than an historical curiosity (see Cummins, 1989, chp.3). But the last few years have seen the beginnings of a seachange in the philosophy of mind. A number of theorists are taking resemblance very seriously again, especially in the form of second order structural isomorphism.[12] And if the line of reasoning that has been developed in this paper is at all on the right track, this group of theorists will soon have its number increased dramatically.

[12] Two theorists who have kept the torch of resemblance, in the form of second order structural isomorphism, burning over the years are Stephen Palmer (1978) and Roger Shepard (Shepard and Chipman, 1970; Shepard and Metzler, 1971). But more recently, James Blachowicz (1997), Robert Cummins (1996), Shimon Edelman (forthcoming), Craig Files (1996), Peter Gardenfors (1996), Daniel Gilman (1994), and Chris Swoyer (1991), have all explored, though in different ways, resemblance relations between representational vehicles and their representational objects.

REFERENCES

Blachowicz, J. (1997): "Analog representation beyond mental imagery", *Journal of Philosophy* 94: 55-84.

Churchland, P.M. (1988): *Matter and Consciousness*. Cambridge, MA: MIT Press.

Churchland, P.S. and Sejnowski, T (1992): *The Computational Brain*. Cambridge, MA: MIT Press.

Cummins, R. (1989): *Meaning and Mental Representation*. Cambridge, MA: MIT Press.

Cummins, R. (1996): *Representations, Targets, and Attitudes*. Cambridge, MA: MIT Press.

Cummins, R. and Schwarz, G. (1991): "Connectionism, computation, and cognition", in *Connectionism and the Philosophy of Mind*, T.Horgan and J.Tienson, eds. Kluwer.

Dietrich, E. (1989): "Semantics and the computational paradigm in cognitive psychology", *Synthese* 79: 119-41.

Edelman, S. (forthcoming): "Representation is the representation of similarities", forthcoming in *Behavioral and Brain Sciences*.

Files, C. (1996): "Goodman's rejection of resemblance" *British Journal of Aesthetics* 36: 398-412.

Fodor, J. (1975): *The Language of Thought*. Cambridge, MA: MIT Press.

Fodor, J. (1992): "The big idea: Can there be a science of the mind?", *Times Literary Supplement* July 3: 5-7.

Gardenfors, P. (1996): "Mental representation, conceptual spaces and metaphors", *Synthese* 106: 21-47.

Gilman, D. (1994): "Pictures in cognition", *Erkenntnis* 41: 87-102

Goodman, N. (1969): *Languages of Art*. Oxford: Oxford University Press.

Hodges, A. (1983): *Alan Turing: The Enigma of Intelligence*. London: Unwin.

Horgan, T. and Tienson, J. (1996): *Connectionism and the Philosophy of Psychology*. Cambridge, MA: MIT Press.

Lewis, D. (1971) "Analog and digital", *Nous* 5: 321-27.

McClelland, J. and Rumelhart, D., eds. (1986): *Parallel Distributed Processing: Explorations in the Microstructure of Cognition Vol. 2: Psychological and Biological Models*. Cambridge, MA: MIT Press.

Palmer, S. (1978): “Fundamental aspects of cognitive representation” in *Cognition and Categorization,* E.Rosch and B.Lloyd, eds. Hillsdale, NJ: Erlbaum.

Rumelhart, D. and McClelland, J. eds. (1986): *Parallel Distributed Processing: Explorations in the Microstructure of Cognition Vol 1: Foundations*. Cambridge, MA: MIT Press.

Sejnowski, T. (1986): “Open questions about computation in cerebral cortex” in McClelland & Rumelhart, 1986, 372-89.

Sejnowski, T. and Rosenberg, C. (1987): “Parallel networks that learn to pronounce English text”, *Complex Systems* 1: 145-68.

Shepard, R. and Chipman, S. (1970): “Second-order isomorphism of internal representations: Shapes of states”, *Cognitive Psychology* 1: 1-17.

Shepard, R. and Metzler, J. (1971): “Mental rotation of three-dimensional objects”, *Science* 171: 701-703

Smolensky, P. (1988): On the proper treatment of connectionism. *Behavioral and Brain Sciences* 11: 1-23.

Swoyer, C. (1991): Structural representation and surrogative reasoning”, *Synthese* 87: 449-508.

Trenholme, R. (1994): “Analog simulation”, *Philosophy of Science,* 61: 115-31.

Van Gelder, T. (1995): “What might cognition be, if not computation?”, *Journal of Philosophy* 91: 345-381.

Von Eckardt, B. (1993): *What is Cognitive Science?* Cambridge, MA: MIT Press.