Computationalism: The Very Idea

David Davenport (david@bilkent.edu.tr)
Computer Eng. & Info. Science Dept.,
Bilkent University, 06533 Ankara –Turkey.

Abstract

Computationalism is the view that computation, an abstract notion lacking semantics and real-world interaction, can offer an explanatory basis for cognition. This paper argues that not only is this view correct, but that computation, properly understood, would seem to be the only possible form of explanation! The argument is straightforward: To maximise their chances of success, cognitive agents need to make predictions about their environment. Models enable us to make predictions, so agents presumably incorporate a model of their environment as part of their architecture. Constructing a model requires instantiating a physical "device" having the necessary dynamics. A program and the computation it defines comprise an abstract specification for a causal system. An agent's model of its world (and presumably its entire mechanism) can thus be described in computational terms too, so computationalism must be correct. Given these interpretations, the paper refutes arguments that purport to show that everything implements every computation (arguments which, if correct, would render the notion of computation vacuous.) It then goes on to consider how physical systems can "understand" and interact intelligently with their environment, and also looks at dynamical systems and the symbolic vs. connectionist issue.

Introduction

Not surprisingly, early digital computers were frequently referred to as "electronic brains." Able to perform complex mathematical calculations, play games and even write simple poetry, these machines clearly displayed characteristics previously thought to be the exclusive realm of human beings. Moreover, they did these feats with such rapidity that it was difficult not to imagine that they would one day exceed our own rather limited cognitive abilities, leading us either into utopia or into slavery.

Of course, this hasn't happened yet (at best, we seem to have a sort of utopic slavery!), but the idea that the mind might be some form of computer has certainly gained considerable currency. Indeed, most research programs in Cognitive Science and Artificial Intelligence adopt this view, known as computationalism –variously the idea that cognition is computation or that minds can be described by programs. Initial success, however, has given way to a series of seemingly insurmountable practical and philosophical problems. These difficulties have prompted some to search for alternative formulations and to question whether computationalism actually has any explanatory value after all.

This paper argues that computationalism, properly understood, is an invaluable tool in our quest to understand cognition. Indeed, it will become obvious that all the alternative proposals are ultimately committed to the very same viewpoint. To appreciate this, however, it is necessary to first develop a sketch of computation and of what a cognitive agent is and how it may function. This perspective will then provide the foundation upon which to sensibly discuss the meanings and relative merits of the various ideas. This approach is somewhat unusual, perhaps, but appropriate given the interrelated nature of the concepts being investigated. Ultimately, these notions simply cannot be understood in isolation, but only as a package, as a coherent philosophy of mind (indeed, a philosophy of everything.[1])

What is Computation?

Consider the relatively intuitive notion of a model. A model is something that stands in for and which "corresponds" in relevant ways, to the system being modelled. The model can thus be used to obtain information about the real system, without the necessity of actually interacting with it. This is clearly advantageous when, for instance, the system is too large to manipulate directly (the solar system, or the weather) or when it might be dangerous or expensive to actually do so (oil spillage from a supertanker or the financial collapse of a major company.) Notice also, that there may be many different ways to model the very same system. Depending on the purpose (the answers required) a model may be more or less accurate, may include or omit various details, and may be implemented in different materials. Models, then, allow for predictions about the way things will be, are, or were, in a given set of circumstances.

Models often bear a physical resemblance to the system they represent. For example, a model yacht is a scaled down version of the real object. Besides being a great play thing, it can be used to answer questions, not just about what the real yacht looks (or would look) like, but about how it might behave in various sea and wind conditions, or how its performance might change if the mast were to be moved slightly forward or aft. Increasingly, however, abstract mathematical/computational models are replacing models that physically resemble the phenomena under investigation.

[1] Davenport (1997), for example, shows how the notion of truth may be analysed using the ideas herein.

One reason for this is the difficulty, time and cost associated with crafting such physical models.

What is a computational model? In essence, it is simply a function[2]; a mapping from input states to output states (which may be thought of as a mapping from questions – including any initial data– to answers.) It is important to emphasize that nothing in this view of computation implies discreteness, i.e. it is perfectly compatible with continuous (real) values. Of course, rather than a single function, the mapping may be specified as the joint (sequential and/or parallel) application of possibly recursive functions. State variables may be used to retain intermediate results between function applications. Computation might also be usefully viewed as a formal axiomatic system (comprising a set of symbols and rules for manipulating them.) These alternative views offer different conceptualisations and allow for different space/time tradeoffs during implementation, however, they do not otherwise add or subtract anything.

Of course, to actually obtain answers, abstract models must be instantiated somehow. This demands a physical implementation, either directly, using, for example, a FSA, or indirectly, by mentally executing the abstraction or using, for example, a digital computer. Implementing an abstract model involves mapping states of the abstraction to physical states, along with any relevant sequences of such states.[3] There are several approaches to achieving this. One might find (or attempt to find) an existing physical entity whose functioning happened to coincide with that desired. This may prove very difficult though, given the constraints of any reasonably complex model.[4] Alternatively, rather than use an existing system, one could construct such a system anew. This then becomes an engineering problem. The thing to note, however, is that in both cases we have to rely on (known, proven) causal[5] interaction to ensure that the desired behaviour is actually forthcoming. Of course, we now have at our disposal a tool –the digital computer– that provides a very quick and easy way to construct an implementation for almost any abstraction[6]. Computer programs are the means by which we specify the desired states and state transitions to our machine. A program (algorithm[7], computation) is thus an abstract specification of the causal behaviour we desire from the system which is to implement our computation (c.f. Chalmers, 1995)

[2] In fact, it may actually be a relation –which admits many-to-many mappings– which may be necessary to properly model concurrent systems (c.f. non-deterministic Turing Machines.)

[3] Where states and "sequences" may be real valued or discrete. Note also that, strictly, there is probably no reason to suppose that the mapping might not be sequence to state and vice versa, much like a Fourier transform. Finally, we might accept to implement a functionally equivalent computation, in which case there need not be an obvious mapping!

[4] Note that the mapping must be repeatable, so that arguments to the effect that one might map such a sequence to anything are unlikely to be fruitful. This point is discussed in some detail below.

[5] Causal may mean nothing more that reliably predictable!

[6] Even Turing Machines are unable to perform some functions, see, for example, Turing (1936) and Copeland (1997).

[7] An algorithm is a sequence of (one or more) function applications. Algorithms comprising different sequences and/or primitive functions may result in the same overall computation. A program is an algorithm in a machine understandable format.

Before moving on to look at cognition and how this view of computation is related to it, it is important to dispose of the argument, put forward by Putnam (1988), to the effect that computation is all pervasive. According to Putnam's proof (in the Appendix of his Reality and Representation), any open system, for example, a rock, can compute any function. If true, this would render void the computationalist's claim that cognition is simply a particular class of computation, since everything, even a rock, would be capable of cognition!

The essence of Putnam's argument is as follows: Every ordinary open system will be in different maximal states, say s_1, s_2, … s_n, at each of a sequence of times, t_1, t_2, … t_n. If the transition table for a given finite state automaton (FSA) calls for it to go through a particular sequence of formal states, then it is always possible to map this sequence onto the physical state sequence. For instance, if the FSA is to go through the sequence ABABA, then it is only necessary to map A onto the physical state $s_1 \vee s_3 \vee s_5$, and B onto $s_2 \vee s_4$. In this way any FSA can be implemented by any physical system.

Fortunately (for cognitive science), the argument is not as good as it may at first appear. Putnam's Principle of Non-Cyclical Behaviour hints at the difficulty. His proof relies on the fact that an open system is always in different maximal states at different times. In other words, it is possible to perform this mapping operation only once (and then probably only with the benefit of hindsight!) But this is of no use whatsoever; for computation, as we have seen, is about prediction. Not only is Putnam's "computer" unable to repeat the computation, *ever*, but also it can only actually make one "prediction" (answer one trivial question.) The problem is that the system is not really implementing the FSA in its entirety. A true implementation requires that the system *reliably* traverse different state sequences from different initial conditions in accordance with the FSA's transition table. In other words, *whenever* the physical system is placed in state s_i it should move into state s_j, and *whenever* it is in s_k it should move to s_l, and so on for every single transition rule. Clearly, this places much stronger constraints on the implementation. Chrisley (1995), Copeland (1996) and Chalmers (1996) all argue this point in more detail. Chalmers also suggests replacing the FSA with a CSA (Combinatorial State Automata), which is like a FSA except that its states are represented by vectors. This combinatorial structure is supposed to place extra constraints on the implementation conditions, making it even more difficult to find an appropriate mapping. While this is true, as Chalmers points out, for every CSA there is a FSA that can simulate it, and which could therefore offer a simpler implementation!

One final point before moving on; Searle (1992) argues that computation is observer-relative, that syntax is not intrinsic to physics, and that his wall might thus be seen to implement the Wordstar program. We have seen that the latter claim is nonsense, however, there is an element of truth in the former ideas. While the relation between the physical implementation and the abstract computation is highly constrained, it is still up to the observer to decide what constitutes a system state and where to draw the boundaries of the system. Clearly, it is also the observer who interprets the system being modelled in terms of the physical states –and thus corresponding abstract computational states. Computation, like beauty, is largely in the eye of the beholder!

What is Cognition?

Agents are usually considered to be small parts of the world in which they exist and are thus assumed to have limited abilities. Cognitive agents are agents that incorporate and use knowledge of the environment to improve their chances of success, even survival!

In order to cope with the vagaries of its world, an agent needs to select and execute the action most appropriate to its goals. This requires making predictions, both about the current state of the un-sensed parts of its world and about the effects of its own actions on the world. Prediction, then, is the principle around which cognition is organised, and an agent's knowledge thus constitutes a model of its environment. The model is a (presumably) physical system that implements a computation capable of providing the necessary answers. The relation between cognition and computation is thus clear.

An agent's model may be innate or it may be constructed (learnt) as a result of sensing and possibly interacting with the environment. It may be static or continuously refined, again as a result of interactions. Given such a model of the world, sensory input must somehow combine with it to determine actions relevant to the agent's present situation and goal. Any discrepancy between the model's predictions and the subsequent sensory input will indicate errors in the model and can thus provide the basis for updating it.

How an agent actually acquires, represents and subsequently utilises the knowledge inherent in its model of the world need not concern us here (see Davenport (1992, 1993) for further ideas in this direction.)

Computationalism

Given the interpretations of computation and cognition outlined above, is computationalism correct? There are at least three ways to interpret this question, (1) Can cognition be described (simulated) by computations, (2) Is cognition literally computation, and (3) Does the notion of computation offer a suitable basis for understanding and explaining cognition.

Based on our analysis, the answer to the first form of the question, "Can cognition be described by computations?" would seem to be "yes." Clearly, we can construct computational simulations of cognition at various levels; the question though, presumably, refers to description at the "lowest" physical-level (if there is any sense to this notion.) Assuming that the mind/brain has a purely physical basis (i.e. no part of it –the soul, perhaps– would continue to exist were its material components to be destroyed) then, since a program/computation is simply a description of a causal system, answering the question in the affirmative requires another physical system having equivalent causal dynamics that we can utilise as the model. This is an empirical problem. It may be the case that the underlying physics of certain cognitive systems do not occur elsewhere. In such a case we might model a specific agent by employing another such agent –if one exists. We could not, however, model the class of such agents in this manner or, equivalently, use the only existing agent to model itself (which would be nonsense –having no predictive value!)

The second form of the question, "Is cognition literally computation?" cannot be answered quite so easily. Computation is certainly part of cognition (specifically, the agent's model of the environment.) But what of the other elements, the input and output pathways linking the model to the environment, the goals, the matching and decision-making mechanism, etc., are they also computational? Again, if they are physical/causal systems, then, presumably, they too can be interpreted computationally, in which case we should also accept that cognition is quite literally a matter of implementing the right form of computational system. Of course, this account does not directly explain the subjective aspects of mind (feelings, desires, etc.) but that is another story.[8]

The final interpretation, "Does the notion of computation fail to have explanatory value when it comes to understanding cognition?" is of more immediate concern to cognitive science and artificial intelligence researchers. Most work in the field tacitly assumes that computation is an appropriate basis, so if this turns out to be wrong there are likely to be massive upheavals! The case against computationalism has been growing stronger of late, with claims to the effect that computation lacks semantics, is disembodied, is insensitive to real-world timing constraints, is at the wrong level, and, most dramatically, that since every system can compute every function, it is just too pervasive to be meaningful.

Clearly, computation is important from a practical perspective and also, perhaps, from a historical one. It has already served as a vital step in the evolution of ideas that will ultimately lay bare the mysteries of cognition. In fact, the case against the computational view of mind is misguided. We have already seen that, while every system can indeed be viewed as implementing some computation, every system simply cannot implement every computation. Moreover, the fact that computation lacks certain elements of mind, such as semantics, is not important, since our objective must be to explain how these features arise. If computation did possess them it certainly could not provide any basis for understanding them! Besides, the notion of a computational model is clearly central to the cognitive process and, at least in the case of semantics, it would appear that we can actually develop explanations in these terms, as the following section explains.

[8] My intuition here is that there is nothing else! Feelings and the like are a natural "by-product" of certain sufficiently complex biological cognitive systems. In any case, we seem to have little choice but to proceed on the assumption that, as Searle (1992) put it, "the mental can be physical too." Note that, even if true, this is not to say that all talk of the mental is redundant. Quite clearly, explanations/models at this level allow for efficiency in our everyday activities. It would be quite impossible for us to function if all "processing" had to proceed at the molecular-level.

Meaning and Representation

What gives some symbols/states meaning? The answer seems obvious, minds do! But this is of no help when the objective is to understand how the mind works. How can mental representations have meaning? AI researchers first suggested that they gained their meaning from other representations. Searle's (1980) infamous Chinese Room Argument was the first nail in the coffin of this idea. Harnad (1993) provided a clearer demonstration of its futility, however, likening the situation to trying to understand the meaning of a word in a foreign language dictionary. Each word is defined in terms of other words, such that, unless someone provides the meanings for a few primitive words, there is no hope of understanding anything!

Obviously, these primitive terms can only acquire meaning as a result of their relation to the world (environment). Attention thus turned to solving this so-called Symbol Grounding Problem. Connectionists saw ANN's (artificial neural networks) as the solution. Others, in particular Harnad, favoured a hybrid approach, whereby a neural network would sit between the environment and a symbol system, isolating appropriate symbols and providing the necessary grounding. Given the apparent limitations of ANN's (lack of compositional structure, etc. as pointed out by Fodor & Pylyshyn (1989), but refuted by later developments, e.g. recurrent neural networks), Harnad's proposal seemed reasonable. On the other hand, the fundamental problem remains. What exactly is the relation between the mental state and the world? Simply "connecting" it (providing a causal pathway) to the environment doesn't exactly resolve this question. Indeed, it probably isn't even a necessary condition. Many alternative explanations, such as co-variance, seem similarly flawed.

Actually, given the analysis of cognition in terms of models, the solution is basically straightforward. A representation (state) has meaning *for the agent* just in case it has predictive value. On relevant occasions the symbol might be activated via causal connections with the environment, indicating that the particular feature it represents is present. On other occasions it may become active as a consequence of the execution of the model and thus constitute a prediction. It may not even have a real-world counterpart, but simply be part of a theory (model), which provides answers in the absence of anything better (the ether or charmed quarks, for instance.) It is not, of course, necessary that the predictions always be correct in order for the state to be counted as a meaningful representation. Neither is it necessary that the agent ever display behaviour based on the representation. This is in contrast to the Interactivist theory of Bickhard and Treveen (1995), which, while similar in other respects, claims that interaction is necessary for real meaning. This seems patently wrong. Few people have "played" with the planets or with electrons, yet we would surely wish to say that they did understand these concepts. If not, education would seem to be a waste of time!

Identifying Symbols & Rules

An agent's model of its world might be viewed as a formal system comprising symbols and inference rules. A number of questions thus arise, first, and foremost of which concerns the origin of these symbols and rules. Are they, perhaps, innate, or does the agent somehow select an appropriate set of symbols? Acquiring (and maintaining) a suitable set of base symbols for a given environment is likely to be one of the primary determinants of success or failure for an agent.

How then, might an agent "discover" the symbols it needs? An outline answer might go something like this. Agents have a number of sensors and actuators. The problem for any agent is to decide which actuator (if any) to invoke at any particular moment. Its objective is to satisfy its needs (food, sex, comfort, etc.) In some cases evolution may have endowed it with automatic (innate) mechanisms that restore it to its "ideal" state. In other situations, however, it will need to instigate "deliberate" actions in the hope of achieving these goals. On the (necessary) assumption that there is some regularity in the environment (and lacking any other prior knowledge), the best an agent can do is to store past sensory input patterns and then match the current situation against these in the hope that they might repeat. The matching process will thus produce a set of expectations, and assuming that the agent has also stored information about its past actions and their effects, it should then be able to compute the "intersection" between these, its perceived situation and its goals, and hence select the most appropriate action to take.

Given the variation in input patterns, the initial problem is to identify sets of sensor inputs that regularly occur together. Having isolated these initial sets, the agent can further group them into less frequently occurring sets, and so on. Gradually, it should also be able to determine combinations of these sets that are mutually exclusive of each other (by observing that they share terms, for example.) All of these groupings form the agent's (internal) symbols. Another set of symbols (external ones) is formed when the agent acquires language. Meaning in these symbols involves an additional mapping from the word itself to the representation of the corresponding concept.

As for the inference rules, they must be basically logical – since the agent must make the correct, rational, "logical" choices. We can thus expect logical rules to be part of an agent’s makeup, i.e. in biological agents, evolution will have produced/favoured mechanisms which behave as if they were performing logical inferences. Classical Logic, being the result of abstraction from our spoken language, is evidence for this, although, of course, it does not account for all our observed reasoning. There have been many attempts to extend Logic (e.g. modal logics, temporal and non-monotonic logics, etc.,) some, however, would argue that the very foundation upon which Logic is built is in need of revision! For some thoughts in this direction, see Kosko (1993) and Davenport (1999). Certainly, human beings frequently fail to reason perfectly (perhaps due to biological limitations, lack of time, incorrect or incomplete knowledge, etc.), but the fact remains that an agent’s mechanism must be inherently logical.

Symbolic, Connectionist or Dynamicist?

Which is the correct view, the symbolic or connectionist one? Or do we need a hybrid solution? Or perhaps another solution altogether, as, for example, the dynamicists would have us believe? It should be clear from the foregoing that, since cognition is described in functional terms, this is essentially an implementation (organisational) issue. Indeed, the same is true of computation. There may be several ways of instantiating the very same computation. They may differ in many respects –energy consumption, speed, reliability, even method[9], etc.– but these are all irrelevant to the computation itself.

In fact, it is far from clear what precisely defines or distinguishes the two archrivals, the symbolic and connectionist paradigms. The symbolic paradigm is usually considered to be closely associated with conventional digital computers whereas the connectionist paradigm is essentially defined by artificial neural network architectures. The situation is not so clear-cut, however. For one thing, ultimately, both can be implemented using the very same technologies, for example, voltages on wires. Another confusion arises from the fact that ANN's can be implemented on (or at least approximately by) symbolic computers and there is reason to believe that the converse is also true. McCulloch and Pitts (1943) purported to show just this, by demonstrating that it was possible to construct logic gates from ANN-like threshold elements. This, however, is not really sufficient since the neural elements are not in practice organised (connected) in this fashion. Some of the limitations on the processing capabilities of connectionist systems, as pointed out by Fodor and Pylyshyn (1989), have been overcome, however, the analysis of this paper clearly points to one remaining shortcoming. Any reasonably sophisticated agent must be able to carry out predictions independent of its current sensor inputs. But most ANN's are simple feed-forward devices and even recurrent ANN's only delay the input one "step." Meeting this predictability requirement would seem to be possible, in principle, although it may demand more complex ANN's, perhaps, ones in which individual neurons retain state information.

So, –assuming that they are both capable of supporting the necessary computational structures– the choice is an organisational one and cognitive agents could equally well employ either. Of course, there may be other reasons to prefer one form to the other. It may be that one is easier to implement in a particular technology; silicon, biology, or beer cans! Or that it requires less hardware or works more reliably.

Given a particular type of agent (e.g. ourselves), it might be useful to be able to determine which form it was employing. Whether this is possible depends on exactly what distinguishes the two paradigms. One point of difference would appear to lie in the mode of storage. In symbolic systems, if a token is to appear in several "expressions" then each will contain a copy of the token. In contrast, the connectionist approach[10] is to retain only one instance of a token, and then to create a link to this instance from each expression in which it participates. It may also be possible to distinguish the paradigms along the lines of how they "view" the world. Symbolic systems often take sets of mutually exclusive tokens (e.g. blue, red, green... or car, bike, truck...) as their starting point, whereas connectionist systems tend to start with (expect) conjunctions of terms. The latter is attempting to identify sets of inputs that always occur together, the former, ones that never occur together. In reality both are impossible to guarantee, so that, as indicated earlier, the only realistic option is for an agent to store input patterns in the hope that they may happen to repeat.

And what of the dynamicist view, is it a viable alternative? Van Gelder (1995) characterises cognitive processes as "state-space evolution within dynamical systems." The archetypical example of a dynamical system he suggests is Watt's centrifugal governor for controlling the speed of a steam engine. Simplifying a little, the basic argument is that the system is not computational since there are no representations or, at the very least, no symbolic representations. From the characterisation of computation given at the beginning of this paper, it will be clear that a system can be computational irrespective of whether or not it displays identifiable symbols. The system clearly has a representation of the current engine speed (via the gearing constant.) Furthermore, since it accurately predicts the amount of steam needed to maintain a steady speed, it must have some representation of the desired engine speed (actually encoded in the mechanical design, weights, arm lengths, angles, etc.) And, of course, while it can be described by complex differential equations, it can also be described by (symbolic) difference equations or, even qualitative language/symbols, although in each case with correspondingly less precision.

Many physical systems do (like the governor) appear to exhibit properties that are not "naturally" separable, i.e. they appear continuous. Conceptually, perhaps, if the universe were (ultimately) comprised of particles and neat slots, it might be possible to agree upon an appropriate set of states (symbols) to employ in describing any system. On the other hand, if, as seems more likely, the universe is not made up of nice discrete entities, then what we decide to call a state is entirely up to us! Indeed, whatever the underlying physics, we as macro agents are continually faced with such situations. We are perhaps fortunate in that most physical entities happen to form relatively static "islands" widely separated from each other. For this reason we find little difficulty in identifying everyday objects, but we quickly begin to falter when faced with gaseous clouds or subatomic particles.

There would thus appear to be no principled way, or reason, to distinguish dynamical systems from any other forms. Indeed, both symbolic and connectionist systems are ultimately dynamical systems. Kentridge (1995) provides a

[9] We may wish to distinguish between the function being computed and the method of computing it. For instance, there are several ways of actually multiplying two values (using repeated addition, shift /add, etc.)

[10] Here we assume the exemplar connectionist system to be a perceptron-like structure that does not employ distributed representations.

good discussion of this showing how rules and differential equations can be equally good descriptions of ANN's.

Timing

Another criticism frequently levelled against computationalism is its failure to say anything about the timing of events. This is because the very notion of computation has traditionally abstracted out time, concerning itself only with sequence. Thus, whether a step in an algorithm takes a second or a millennium doesn't matter, the end result will (must) be the same. The increasing use of GUI's and embedded systems, however, seems to make the traditional (Turing Machine inspired) view of computation, –as a purely formal system in which each state is always followed by the very same next state– rather less appropriate. Today, computer systems must handle asynchronous input from the environment and respond in a timely manner. So, the question is whether it is necessary to complicate the classical picture of computation with actual time values or whether pure sequences can still suffice.

Obviously, a system has to be fast enough to cope with the demands made upon it, otherwise it will fail to "keep up." Every technology has its limits though, biological ones perhaps more so than others, so there will always be situations that an agent constructed with a given technology will be unable to handle. Engineers, asked to design/build a system guaranteed to cope with events having certain temporal characteristics, need to be concerned with the actual minimal response times of their system, and may, perhaps, choose a faster technology if necessary. On the other hand, if the technology is fixed, as for example is our own, then there is little more that can be done (after selecting an optimal algorithm.)

Another possible concern relates to situations in which the agent may respond too quickly and hence fail to achieve its goal. While this may be solved by the addition of timing information, (e.g. explicit time delays) it might also be handled by finding conditions for invoking the actuator that are more specific. In reality, this might include taking inputs from naturally occurring "clock" signals (e.g. the human heart beat); however, this would not constitute timing information, per se.

The final reason that timing might be important, relates to stability concerns. Avoiding oscillation in control systems is clearly important and the development of the mathematical tools necessary to guarantee this has been one of the major achievements of modern control theory. Unfortunately, even in the case of relatively simple linear systems, the analysis is very complex, and there would seem to be no natural way to extend it to non-linear systems many orders of magnitude more complex. The only hope would seem to be systems that could learn and adapt so as to automatically stem oscillation as much as possible.

In our quest to understand cognition, extending the notion of computation to include timing information thus seems unnecessary. On the other hand, techniques that would allow performance criteria to be evaluated would certainly be beneficial.

Summary & Concluding Remarks

Does computation, an abstract notion lacking semantics and real-world interaction, offer a suitable basis for explaining cognition? The answer would appear to be yes, indeed, it would seem to offer the only possible explanation!

The basic argument of this paper is as follows. Models enable us to make predictions. Constructing a model requires building a physical "device" whose states and dynamics map onto those of the target system. A convenient way to do this is to write a program that can be executed on a digital computer. The program, and the computation it defines, is thus an abstract specification of the desired causal system. To maximise their chances of success, cognitive agents need to make predictions about their environment. It therefore seems reasonable to assume that their architecture must include a model that can be used to make such predictions. This model can be described and interpreted in computational terms, so computationalism must offer an appropriate basis for explanation.

While connectionists and dynamicists claim to offer alternative models, it is clear that these relate to organisational concerns and thus do not deflect the essential computational explanation, for they too are computations! The argument put forward by roboticists, psychologists and social theorists, that intelligence/representation demands situated interaction, would appear to be essentially correct on the analysis presented here. A state is representational only on the basis of its predictive value to the agent. From the computational viewpoint this is perfectly natural and answers the question of semantics. Finally, the philosophical argument, which claims to show that computation is a potentially vacuous concept, was seen to be misleading. Mapping every computation to every system is simply not possible because the proper causal structure is lacking. Computation is about prediction and while it is possible to map any specific computational sequence onto (almost) any physical system, there is little predictive value in so doing.

Using these ideas we might envisage a hierarchy of systems based on their cognitive abilities. At the bottom would be FSA-like machines that have no inputs or outputs. Above them, purely reactive systems with a few inputs and outputs but fixed causal pathways and no internal state. Next would come adaptive systems, slightly more sophisticated, being able to modulate their behaviour within the limits of their fixed physical structure. These may be followed by FSA that do have inputs and outputs. Above these would come systems like the modern digital computer and the Turing Machine that have I/O, a significant number of states and a practically infinite ability to reconfigure their causal structure. Lastly, at the top of the hierarchy we might place systems that not only have the ability to rewire themselves, but also to expand the number of inputs, outputs and states available to them.

A major objective of this work was to establish a simple coherent framework within which to understand the notions of computation and cognition, and the relation between them. By taking a broad view of computation and examining what it is to implement one, we have hopefully made progress in a way that respects the mass of existing theoretical work and yet retains our intuitions.

References

Bickhard, M. H. & Terveen, L. (1995). *Foundational Issues in Artificial Intelligence and Cognitive Science: Impasse and Solution.* Advances in Psychology 109, North-Holland, Elsevier.

Chalmers, D. J. (1995). On Implementing a Computation. *Minds and Machines 4,* pp.391-402.

Chalmers, D.J. (1996). Does a Rock Implement Every Finite-State-Automaton? *Synthese, Vol.8 No.3*, pp.309-333, Kluwer Academic Pub.

Chrisley, R.L. (1995). Why Everything Doesn't Realize Every Computation. *Minds and Machines 4,* pp403-420, Kluwer.

Copeland, B.J. (1996). What is Computation? *Synthese, Vol.8 No.3*, pp.335-359, Kluwer Academic Pub.

Copeland, B.J. (1997). The Broad Conception of Computation. *American Behavioral Scientist, Vol.40, No.6*, pp.690-716, Sage Pub.

Davenport, D. (1992). Intelligent Systems: the weakest link? In Kaynak, O., G.Honderd, E. Grant (1993) *Intelligent Systems: Safety, Reliability and Maintainability Issues*, Berlin: Springer-Verlag.

Davenport, D. (1993). Inscriptors: Knowledge Representation for Cognition. *Proceedings of Eighth Int. Symposium on Computers and Information Science,* Istanbul.

Davenport, D. (1997). Towards a Computational Account of the Notion of Truth. *Proceedings of the 6th Turkish Artificial Intelligence and Neural Network Symposium.*

Davenport, D. (1999). *The Reality of Logic.* A talk to the Cognitive Science Group, Middle East Tech. University. (see http://www.cs.bilkent.edu.tr/~david/david.html)

Fodor, J. & Pylyshyn, Z., (1989). Connectionism and Cognitive Architectures: A Critical Analysis. In Pinker, S. & Mehler, J., (Eds.) (1990). *Connections and Symbols (A Special Issue of the Journal Cognition),* Bradford Books, MIT Press.

Harnad, S. (1993). Grounding Symbols in the Analog World with Neural Nets. *Think* (Special Issue on Machine Learning.)

Kentridge, R.W. (1995). Symbols, Neurons, Soap-Bubbles and the Neural Computation Underlying Cognition. *Minds and Machines 4,* pp439-449, Kluwer.

Kosko, B. (1993). *Fuzzy Thinking: The New Science of Fuzzy Logic.* New York: Hyperion.

McCulloch, W.S. & Pitts, W. (1943). A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics,* 5, pp.115-133.

Putnam, H. (1988). *Representation and Reality*. Cambridge, MA: MIT Press.

Searle, J. (1980). Minds, Brains and Programs. *Behavioural and Brain Sciences 3*, pp.417-424. Reprinted in Boden, M. (Ed.), (1990) *The Philosophy of Artificial Intelligence*, Oxford Univ. Press.

Searle, J. (1992). *The Rediscovery of the Mind*. Cambridge, MA: MIT Press.

Turing, A.M. (1936). On Computatble Numbers, with an Application to the Entscheidungsproblem. *Proc. Of the London Mathematical Society, Series 2, 42*, pp.230-265.

van Gelder, T. (1995). What Might Cognition be, If Not Computation. *The Journal Of Philosophy*, Vol.XCI, No.7