Turing's Analysis of Computation and Theories of Cognitive Architecture

A. J. WELLS
The London School of Economics and Political Science

Turing's analysis of computation is a fundamental part of the background of cognitive science. In this paper it is argued that a re-interpretation of Turing's work is required to underpin theorizing about cognitive architecture. It is claimed that the symbol systems view of the mind, which is the conventional way of understanding how Turing's work impacts on cognitive science, is deeply flawed. There is an alternative interpretation that is more faithful to Turing's original insights, avoids the criticisms made of the symbol systems approach and is compatible with the growing interest in agent-environment interaction. It is argued that this interpretation should form the basis for theories of cognitive architecture.

I. INTRODUCTION

Turing's analysis of the process of routine computation (Turing, 1936–7), which included the construction of an abstract universal computing machine and a proof that there are uncomputable functions of real numbers, is one of the outstanding intellectual achievements of the twentieth century. Turing's work provides the foundation on which theoretical computer science has been built and forms part of the "deep background" of cognitive science. In this paper it is argued that Turing's analysis of computation is of more than background significance because it is directly relevant to contemporary debate about theories of cognitive architecture.

The dominant paradigm in cognitive science from the late 1950's until the early 1980's viewed the mind as a symbol system and was based on a model drawn from the technology of digital computers. Since then, a variety of other approaches have challenged this paradigm. Connectionism, situated action theory, artificial life and dynamical systems theory among others, all cast doubt on the central claim that cognition consists of the manipulation of internal symbol structures. Some have even questioned the adequacy of the paradigm of
computation developed by Turing. Smolensky (1988, p. 3), for example, suggested that connectionist models might “challenge the strong construal of Church’s Thesis as the claim that the class of well-defined computations is exhausted by those of Turing machines.”

Although there has been a sustained and vigorous counterattack by symbol systems theorists, (e.g. the much cited paper by Fodor & Pylyshyn (1988) and the debate which followed it) a body of opinion now exists in cognitive science that rejects computational ideas altogether. Clark has called this view “The Thesis of Radical Embodied Cognition”. According to this thesis, “Structured, symbolic, representational, and computational views of cognition are mistaken. Embodied cognition is best studied by means of noncomputational and nonrepresentational ideas and explanatory schemes involving, e.g., the tools of Dynamical Systems theory.” Clark (1997, p. 148). Clark does not himself deny the value of computational thinking or the need for it, but he does argue that we need a fundamental overhaul of our approach to understanding the mind. Part of that overhaul, he suggests, should involve close attention to the concept of computation. It is, he says, “one of the scandals of cognitive science that after all these years the very idea of computation remains poorly understood.” Clark (1997, p. 159). Hutchins (1995a,b), whose work broadens the unit of cognitive analysis from the individual to distributed socio-technical systems, has also argued for a review of the relation between symbol processing and theories of cognitive architecture. “I believe that humans actually process internal representations of symbols. But I don’t believe that symbol manipulation is the architecture of cognition. Historically, we simply assumed that symbol processing was inside because we took the computer as our model of mentality.” Hutchins (1995a, p. 370).

Both Clark and Hutchins endorse an approach to cognition that emphasizes the interactions between the brain and the external environment. The point is not just to stress that humans routinely make use of external aids to supplement short term memory limitations or use tools to extend their physical and cognitive capacity. This type of idea goes back at least to Newell and Simon’s (1972) analysis of the contribution of task environments in their work on human problem solving. What Clark, Hutchins, and many other theorists are arguing for is a fundamentally altered view that replaces the idea that the brain is an autonomous symbol processor with the idea that the brain is first and foremost the controller of embodied activity. The challenge for such an approach, when it does not reject computation tout à fait, is to come to an appropriate understanding of it.

The contention of this paper is that Turing’s analysis of computation has a fundamental role to play in this project because it provides just the clear understanding of computation that Clark suggests is scandalously lacking. Moreover, the interpretation of Turing’s work that is outlined here supports the change of approach espoused by Clark, Hutchins, and others. What is needed is not a new paradigm of computation but a better understanding of the existing one. The problem, I shall argue, is not that the idea of computation is poorly understood, but that the appropriate interpretation of the idea that is needed to underpin theories of cognitive architecture has not been grasped. Turing provided the right ideas with his formal analysis of computation, but cognitive scientists in the symbol processing tradition have based their theories on the technology of digital computers, which employs a particular, but not the only possible, realization of Turing’s machine model.
The Turing machine has two principal parts, a finite state controller and a tape. The digital computer also has two principal parts, a central processor and a memory. The central processor is the functional equivalent of the finite state controller of a Turing machine, and the memory is the functional equivalent of the tape. Symbol systems theorists argue that the human brain functions in the same way as a computer, or, equivalently, as a Turing machine. Thus, by hypothesis, there must be mechanisms in the brain that are functionally equivalent to the finite state controller/central processor and to the memory/tape. Symbol systems theorists pay particular attention to the memory/tape mechanisms because they are hypothesized to contain structured symbolic representations of the world. The manipulation and transformation of these representations constitute cognitive activity. This approach adopts what is here called the internalist interpretation of the Turing machine architecture because it assumes that both the principal parts of the TM are instantiated in the brain. The internalist interpretation is essentially what has been espoused by generations of theorists including Miller, Galanter, and Pribram (1960); Newell and Simon (1960, 1972, 1976); Newell (1980, 1990); Fodor (1975, 1983, 1994); Pylyshyn (1984); Fodor and Pylyshyn (1988); Vera and Simon (1993) and Pinker (1994). Part of what has made the internalist interpretation seem compelling is that it is derived from the remarkably successful technology of stored program, digital computers. These successes tend to mask the difficulties into which the internalist interpretation leads theories of human cognitive architecture. It is argued that these difficulties are more formidable than is commonly recognized.

Although the internalist approach has grounded the work of many prominent theorists of cognitive architecture, it is not the only interpretation of the Turing machine on which a theory of cognitive architecture can be based. This paper argues for an alternative approach in which only one of the principal parts, the finite state controller/central processor, is hypothesized to be instantiated in the brain. The other principal part, the memory/tape mechanism, is hypothesized to exist in the external environment. Consequently, cognitive computation is a process of organism-environment interaction. This approach is based on what is called the interactive interpretation of the Turing machine architecture. It might at first sight appear that an immediate and fatal objection to this approach is that it shifts the locus of cognitive representations from the organism to the environment. This is not so. The argument is not that cognitive representations are external to the organism, but that they are instantiated in the states of the finite state controller/central processor rather than as symbolic structures in a putative neural memory/tape mechanism. It is argued that this approach is more faithful to the thinking which led to the Turing machine in the first place and also that it avoids the serious problems from which the internalist approach suffers. The interactive interpretation of the Turing machine provides a discrete state approach to the principles of interaction which are currently being studied in continuous form by proponents of dynamical systems theory (e.g., Beer, 1995). When looked at in this light, Turing's analysis of computation can be seen as the first systematic exploration of agent-environment interactivity, a topic currently of interest in Artificial Intelligence (cf. Agre, 1995). It is often said of Turing that he had many ideas which were ahead of their time. His work on programming in the late 1940's and his work on the chemical basis of morphogenesis in the early 1950's are well known instances. What I think we will come to see is that
his work on the theory of computation in the 1930's is a precursor of the interactive paradigm, which will take cognitive science and AI into the twenty first century.

The paper is structured as follows: Section 2 reviews the Turing machine formalism. Section 3 describes the relation between the Turing machine and digital computer architectures. Section 4 outlines the symbol systems hypothesis. Section 5 describes two types of problems arising from the architecture of symbol systems. Section 6 outlines the interactive interpretation and grounds it in a detailed exegesis of Turing's work. Section 7 indicates some areas where the interactive interpretation is likely to be fruitful. Section 8 draws conclusions.

II. THE TURING MACHINE

In the early summer of 1935, at about the time when Turing began work on the paper on computable numbers, Hodges (1983, p. 96), the world was innocent of electronic, digital computers. Computation was something done by people, most often working with pencil and paper. "The computer" was not a machine but a person carrying out a computation. Turing wanted to answer the question "What are the possible processes which can be carried out in computing a number?" Turing (1936-7, p. 135). To this end he developed the idea of an abstract machine, now called the Turing machine (hereafter TM), which modelled the processes carried out by the aforesaid human computer. He was not, it is crucial to note, modelling just the processes going on inside the mind of the computer, but the interactive processes involving both the mind and the external medium, that is paper, which was used for intermediate workings and the recording of results. Turing suggested that the mind of the (human) computer could be modelled by "a machine which is only capable of a finite number of conditions" Turing (1936-7, p. 117) and that the paper on which the (human) computer worked could be modelled as a paper tape "divided into squares like a child's arithmetic book" Turing (1936-7, p. 135). The squares of the tape contained symbols. The complex of perceptual and motor systems whereby the (human) computer reads and writes symbols on paper is modelled by a notional connection between the tape and the finite machine. Turing thought of the tape as "running through" the machine in such a way that a single square at a time is "in the machine". He called the square in the machine the "scanned square" and the symbol on this square the "scanned symbol". The finite machine can move relative to the tape so as to change the scanned square. Modern accounts of the TM architecture refer to a read/write head, which carries out the needed tasks. Because the finite control models the mind of the human computer and the tape models the paper on which the human writes, the TM as a whole is a model of an interactive system consisting of an agent and an environment.

Every TM has the same set of notional "parts": a finite control machine, a tape (which is infinite in principle) and a read/write head which connects them. What distinguishes one TM from another is the finite set of "conditions" or "internal states" of the machine, the way in which they are organized, the alphabet of symbols which the machine uses and the form of the input strings which it requires. I propose to use a number of different terms to distinguish different aspects of the TM architecture. I shall call the invariant set of parts
TURING AND COGNITIVE ARCHITECTURE

(i.e. the control, tape, and read/write head) the "structural architecture". The particular set of internal states of a given TM, its symbol alphabet, and the syntactic constraints on its input strings, I shall call the "task architecture", because they specify the task carried out or function computed by the machine. The task architecture has two distinct but interacting parts. One is the system of internal states of the machine. I shall call this the "control architecture". The other is the set of syntactic constraints which specify the allowable forms of input strings. I shall call this the "input architecture".

Taking the structural architecture as given, the task architecture of a TM can be defined more formally as a quadruple \((Q,S,D,\phi)\) where \(Q = \{q_i \mid 1 \leq i \leq m\}\) is a set of internal states not including the halt state \(h\), \(S = \{s_j \mid 0 \leq j \leq n\}\) is an alphabet of symbols including the blank symbol \# and \(D = \{L,R,N\}\) is a set of movement indicators. \(q_1 \in Q\) is the starting state of the machine and \(\phi\) is a function from \(Q \times S\) to \(S \times (Q \cup \{h\}) \times D\). The set of arguments and values of \(\phi\) specifies the "machine table" of a given TM. If \(\phi(q_i, s_j) = (s_p, q_p, d_i)\) then the machine when in state \(q_i\) and scanning symbol \(s_j\) will rewrite \(s_j\) as \(s_p\), enter state \(q_p\), and move in direction \(d_i\). The machine halts only when \(q_s = h\) or when the value of \(\phi(q_i, s_j)\) is undefined.

As an example, the machine table for a simple TM is shown in Table 1. \(Q = \{q_1, q_2, q_3, q_4\}\), \(S = \{\#,0,1,X,Y,A\}\). The machine computes the product of a pair of positive integers expressed in unary notation. The input to the machine is in the form of a string of symbols \(Y1^*X1^*X\), where \(1^*\) represents zero or more 1's. The first string of 1's represents the multiplicand and the second the multiplier. Thus \(Y11X11X\) represents the input \(2 \times 2\). The product is written to the left of the "Y". The expression \(2 \times 2 = 4\), for example, is represented by the string \(1111Y11X11X\). The machine is started in state \(q_1\) scanning the "Y" at the left end of the input string.

A crucial feature of the relationship between the control architecture and the input architecture of every TM is that they are co-designed. The control architecture is designed with a specific input architecture in mind and vice versa, and the function computed is an emergent property of their interaction. Typically, it is impossible to tell what the function is by studying either the control or the input architecture alone. If the organization of either is changed a different function is computed, or no function at all. The relationship between control architecture and input architecture is a central concern of the interactive interpretation of the TM discussed in Section 6, but has rarely been discussed by theorists committed

<table>
<thead>
<tr>
<th>1</th>
<th>A</th>
<th>X</th>
<th>Y</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Q_1)</td>
<td>1,q_1,R</td>
<td>A,q_3,L</td>
<td>X,q_2,R</td>
<td>Y,q_1,R</td>
</tr>
<tr>
<td>(Q_2)</td>
<td>A,q_4,L</td>
<td>A,q_2,R</td>
<td>X,h,−</td>
<td></td>
</tr>
<tr>
<td>(Q_3)</td>
<td>A,q_4,L</td>
<td>A,q_3,L</td>
<td>X,q_3,L</td>
<td>Y,q_4,R</td>
</tr>
<tr>
<td>(Q_4)</td>
<td>1,q_4,L</td>
<td>1,q_4,R</td>
<td>X,q_2,R</td>
<td>Y,q_4,L</td>
</tr>
</tbody>
</table>

Note. The first row of the table lists the alphabet of symbols, the first column the states of the machine. The cells in the body of the table list the actions the machine takes for the given combination of state and symbol.
to the internalist interpretation. Attention has largely been focused on the consequences for
cognitive theory of one of Turing's central achievements, the design of a "universal"
machine.

Machines like the multiplier of Table 1 compute the single function determined by their
task architecture. Such machines are deterministic by design. Because of this, tracing the
course of a computation is itself a rule governed, deterministic process and Turing saw that
a machine could be designed to carry it out. He called this machine the "universal"
machine. The control architecture of the universal machine was designed to take the
machine table of another Turing machine M as its input, and to produce the output that M
would have produced. It worked in essentially the same way as you or I work if we take the
machine table of a TM and work through a computation step by step.

The impact of the universal Turing machine concept has been profound. It showed that
it was possible in principle for a single mechanism to carry out an infinite number of dif-
ferent tasks. The machine U had the same structural architecture as any other TM and a
fully specified task architecture, but it was capable of an infinity of different tasks because
its input architecture took the form of the machine table of another TM and its control
architecture was designed to interpret that machine table. The function computed could be
changed by substituting one machine table for another as input. The universal machine is,
in fact, an abstract example of a programmable machine and demonstrates the close linkage
between the concepts of programmability and universality. The universal machine concept
also showed that appropriately structured machines were, in principle at least, capable of
an astonishing degree of behavioral flexibility.

III. TURING MACHINES AND DIGITAL COMPUTER
ARCHITECTURE

Although the Turing machine formalism has a very physical flavor, the arrangement of
tape, control and read/write头 which constitutes the structural architecture was purely
notational. It was theoretically impeccable and served as an ideal vehicle for Turing's ideas
about the task architecture needed for universal computation, but it had little to recommend
it as the basis for building a practical computing machine. In any case, the mainstream of
electronic, digital computer development during the 1940's owed very little to Turing's
ideas. For example, von Neumann certainly knew of Turing's work, but his highly influen-
tial document on the design of the EDVAC, which established the principles of what
became known as the von Neumann architecture (cf. Aspray & Burks, 1987), made no ref-
erence to the Turing machine. It was largely in retrospect that the connection between prac-
tical computers and Turing's abstract machines became understood. The focus of
discussion here, however, is not historical but theoretical, and the purpose of this section is
to indicate the relationship between the architecture of the Turing machine and the archi-
tecture of a generic, digital computer.

Theoretically, a stored program, general purpose, digital computer is a practical version
of a universal Turing machine in the sense that it has the task architecture of a universal
machine mapped onto a more practical, structural architecture. Turing himself wrote an
illuminating paper about the issues in 1947 (reprinted in Carpenter & Doran, 1986) in which he likened the access problems created by a memory like the tape of a TM to the problems the Egyptians presumably had with papyrus scrolls. The problem of building suitable memories was the single most important challenge facing the pioneers of digital computing. In the von Neumann architecture the infinite tape memory of the TM becomes a suitably large but finite electronic memory (supplemented where necessary by external media such as floppy disks, magnetic tapes etc.), the control automaton of the TM becomes the CPU of the von Neumann machine and the addressing mechanism of the TM, i.e. the read/write head and movement relative to the tape is replaced by hard wired connections between the CPU and memory and the use of address arithmetic to determine the location of instructions and operands. In the von Neumann computer, the task architecture of the universal TM with its essential distinction between control architecture and input architecture is preserved in a structural architecture which no longer maps those distinctions as directly as the TM architecture does. It becomes extremely easy, in consequence, to lose sight of the fact that Turing’s distinction between control and tape enshrined a distinction between an agent and an environment.

The structural architecture of the digital computer has to extend beyond the mapping of the task architecture of the TM just described, however, in order to be suitable for practical computation. In addition to the parts discussed, peripheral equipment for input and output is needed. This is provided in personal computers by devices such as the keyboard, mouse and VDU. Thus in a practical digital computer, the task architecture of a universal TM is mapped onto the internal components of the computer and this internal package is then embedded in the peripheral equipment which makes its functionality available to the user.

### IV. THE ARCHITECTURE OF SYMBOL SYSTEMS

For the purposes of thinking about theories of cognitive architecture and their relation to Turing’s analysis of computation, the most important aspect of the architecture of digital computers described above is the way in which the structural distinction between the finite state control and the infinite tape memory of the Turing machine is weakened. Digital computers maintain the functional distinction between control and memory, but blur the structural distinction by hardwiring the control to a finite memory and packaging the resulting system into a single box. This blurring has three notable consequences. First, it transforms what was an interaction between an active control mechanism or agent and a qualitatively different external medium in the TM into an interaction between different parts of one internal system. Second, it provides prima facie plausibility, given the existence of working computers, for the idea that mental functioning might result from a similar functional distinction implemented somehow in the structural architecture of the brain. Third, it underpins the idea that mental computation is internal symbol processing. These three ideas lead readily to the hypothesis that the apparently limitless behavioral and cognitive flexibility of the human mind results from the fact that the structural architecture of the brain (i.e. its hardware), implements the task architecture of a universal machine. This is the basis of the Physical Symbol Systems Hypothesis (Newell & Simon, 1976) and (New-
ell, 1980). One particularly telling feature of the hypothesis is that it restores the concept of mental representation to a central place in the theory of cognition. Universal machines rely essentially on representations. Thus the postulate that the mind is a universal computer implemented in neural tissue explains behavioral flexibility in terms of the highly desirable, but previously suspect, notion of mental representation.

The core idea that the architecture of the mind is functionally equivalent to a TM architecture embedded in peripheral systems that communicate with the external world is made explicit by, among others, Newell (1980, 1990); Fodor (1983); and Pylyshyn (1984). The literal commitment to internal symbol processing and the requirement for structures and processes that transform physical stimuli into symbolic codes are features of all these works. The requirement for syntactically complex symbol structures is a central feature of the attack on connectionism mounted by Fodor and Pylyshyn (1988). The central theoretical importance of internal symbol processing appears, in fact, to be agreed by all proponents of what is often called the computer model of the mind. The degree of commitment to the notion that the internal architecture is computationally universal is slightly less clear.

Universality requires that most programs must be represented explicitly. Newell and Simon (1972) argued that the notion of programmability was the key to understanding human behavioral flexibility and programmability implies universality. By the time they wrote their Turing award paper, Newell & Simon (1976), the commitment to universality as a criterial feature of the internal task architecture had become explicit and remained so in later works (eg., Newell, 1980, 1990). The commitment is less clear in the case of the “classical” theorists. Fodor and Pylyshyn, for example, argue that their conception of cognitive architecture allows, but does not require, programs to be encoded explicitly as symbol structures. Data, by contrast, they claim must be explicitly encoded: “What does need to be explicit in a Classical machine is not its program but the symbols that it writes on its tapes (or stores in its registers). These, however, correspond not to the machine’s rules of state transition but to its data structures.” (Fodor & Pylyshyn, 1988, p. 61). What is not clear is whether Fodor and Pylyshyn think that any of the mind’s programs are explicit or whether they consider all the programs to be implicit (i.e., wired in). If the latter is the case, then clearly the cognitive architecture cannot be universal, because only a finite number of programs could be wired into a finite system like a brain. If programs sometimes are explicit, however, then there must be a program interpretation mechanism somewhere in the architecture; thus, subject to constraints of time and space, the architecture can be considered universal. In other work, Fodor and Pylyshyn both write in ways that make the latter construal more plausible. They also refer to Newell’s work as part of the Classical approach. Therefore, I take it that internal symbol processing and computational universality (or at least substantial programmability) are both criterial features of the symbol systems interpretation of the Turing machine model.

V. PROBLEMS WITH THE ARCHITECTURE OF SYMBOL SYSTEMS

The fundamental claim of the symbol systems approach to cognitive computation as discussed above is that the mind has the task architecture of a universal machine realized in
the structural architecture of the brain. This claim has the two following corollaries: first, the mind is a programmable architecture, and second, mental computations are computations over internal symbol structures that represent the external world. Each of these corollaries leads to a serious difficulty. The claim that the mind has a general-purpose, programmable architecture conflicts with the fact that the mind is a product of evolution by natural selection, and the claim that the mind computes over internal symbolic representations encounters what I shall call the “transduction problem”. These difficulties suggest that the mind is not a symbol system.

The Evolutionary Problem

The first argument presented here is based on the work of Michael Conrad. Conrad argues that there is a trade-off principle that shows that computational efficiency and evolutionary adaptability are incompatible with programmability. “A system cannot at the same time be effectively programmable, amenable to evolution by variation and selection, and computationally efficient. The von Neumann computer opts for programmability. The trade-off theorem suggests that an alternative domain of computing is in principle possible, where programmability is exchanged for efficiency and adaptability. Biological systems, as the products of evolution, must operate in this alternative domain.” (Conrad, 1985, 464–5, italics in original). The essence of the argument is that programmability imposes constraints on the structure of a computational system that are incompatible with the modifiability required for evolution.

Conrad argues that evolutionary adaptability (i.e., the capacity to survive in uncertain environments as a result of the operation of variation and selection mechanisms), requires both evolutionary flexibility and what he calls gradual transformability. Evolutionary flexibility is essentially a measure of the capacity of a system to undergo structural changes, while continuing to be able to solve a particular class of adaptive problems. The greater the number of such structural changes that a system can sustain, the greater its evolutionary flexibility. The types of structural change that are relevant depend on the system under consideration. Conrad mentions single mutations or crossover events in DNA and single alterations of code in computer systems as examples.

The concept of gradual transformability links structural changes to a system with changes in its behavior, again taken relative to a particular class of adaptive problems. Structural changes are likely to be sustainable if they result in small rather than large changes in behavior; Conrad defines a system as being “gradually transformable” if it undergoes only a small change in behavior as a result of a structural change. The question then is what constitutes a small change in behavior. This question has no general answer, but Conrad suggests that in the context of evolution it is reasonable to think of the behavior of two systems as differing by a small amount, if each solves approximately the same class of adaptive problems or performs adequately in approximately the same set of environments. A weak definition of adequate performance for a Turing machine or computer program is that its computation should terminate. Programs in infinite loops are almost never doing what is required. Conrad uses this idea to define a small change in behavior:
"[A]ll systems which perform different but defined computations differ by a small amount as long as their computations terminate. We will say that two systems whose behavior differs from each other in this weak sense of small are weakly transformable into one another if they can be transformed into one another by a sequence of elementary structural changes. Weak transformability can be taken as a necessary condition for gradual transformability." (Conrad, 1988, p. 294, italics in original).

Conrad shows that programmable systems do not satisfy even this very weak notion of gradual transformability. His argument leads to the conclusion that the class of programmable systems fails to meet a necessary condition for evolutionary adaptability, and this suggests that minds are not programmable systems in the way in which digital computers are, given that minds are the products of an evolutionary process.

A different form of evolutionary argument can be found in the work of Tooby and Cosmides (1992). This argument focuses on the relation between evolution and behavioral flexibility. Newell (1990, p. 61), suggested that the plasticity afforded by computational universality is just what is required to explain mental capacity because it "opens up the whole world of indefinitely rich representations". Tooby & Cosmides suggest, to the contrary, that evolution could not have selected for the arbitrary undirected behavioral plasticity that is characteristic of universal computers. The problem is that there is an infinite number of ways in which a biological system organized as a universal computer could act, the majority of which would be lethal within a relatively short period. In consequence, they argue, "to be endowed with broad behavioral plasticity unconnected to adaptive targets or environmental conditions is an evolutionary death sentence, guaranteeing that the design that generates it will be removed from the population." Tooby & Cosmides (1992, p. 101).

Since broad behavioral plasticity is the primary functional attribute of stored program computers, this evolutionary argument casts doubt on the suggestion that the task architecture of the mind could be organized in this way.

The Transduction Problem

Another major problem for the symbol systems approach to cognitive architecture is characterized here as the "transduction problem". It is tempting to call it "Pylyshyn's problem" because the issues are very clearly identified by Pylyshyn (1984). The background to the problem is this: If the structural architecture of the brain supports the functional architecture of a universal Turing machine, then there is a requirement for all the data over which cognitive computations operate to be presented in symbolic format. It is the responsibility of what Pylyshyn calls a transducer to meet this requirement. We can understand what is involved, roughly, by considering what the peripheral equipment attached to a computer does. It translates input from the external world in the form of keypresses or button clicks, for example, into appropriate symbol structures in memory. Peripheral input systems are the only way in which a computer acquires data; that is to say, they are the only channel between the internal functional architecture and the external world. Pylyshyn recognizes that the architecture of symbol systems imposes a similar constraint on the human cognitive system. In this case it is the input transducers that have to do the work. He is admirably clear about the force of the constraint. "Because the output [of the transducer] is to serve as
the basis for the only contact the cognitive system ever has with the environment, it should provide all (and only) cognitively effective information." Pylyshyn (1984, p. 158). Now it is evident that one primary difference between the human cognitive system and a computer is that human cognitive data are not provided by an external agent in the way in which the user provides data for a computer. Consequently, it is incumbent on a human transducer to filter its input so as to extract what the cognitive system needs from the indefinitely complex stimulus world impinging on it. The transducer must, in other words, make a judgement of cognitive relevance. Pylyshyn is also clear about this requirement: "...the output of the set of transducers available to an organism must preserve all distinctions present in the environmental stimulation that are also relevant to the explanation of some behavioral regularity." Pylyshyn (1984, p. 158, italics in original). The problem arises because relevance is a cognitive property but transducers are pre-cognitive and, according to Pylyshyn, heavily constrained. Their function is non-symbolic otherwise they would be part of the internal cognitive architecture, they are primarily stimulus bound and their output is a stream of atomic symbols or n-tuples.

Consequently, requiring that the output of transducers should respect some criterion of cognitive relevance appears to be requiring something that is beyond their capacity in principle. It seems analogous to requiring that a computer keyboard filter its input according to some criterion of relevance. Pylyshyn is absolutely correct, I believe, to state the constraints on transducers as he does. They appear to follow directly from the assumption that the cognitive architecture is a symbol system embedded in a matrix of peripheral non-cognitive systems. Where he errs is in thinking that the transducer functions he describes are possible. "Pylyshyn's problem", in a nutshell, is that his characterization of transducer functions constitutes an argument against the possibility of the cognitive architecture they are intended to serve. It is curious that although Pylyshyn identifies the relevance constraint, he never indicates how it might be met by a transducer operating under the other constraints he identifies.

A rather similar criticism of transduction is made by Bickhard and Terveen (1995) as part of a much wider critique of a range of cognitive scientific theories. Bickhard and Terveen argue that theories which assume that representation is constituted as some form of encoding are fundamentally incoherent because they cannot give an account of the emergence of representations with epistemic properties from purely causal foundations. Bickhard and Terveen apply their critique to a wide range of theories including the symbol systems approach.

VI. INTERACTIVE COGNITIVE ARCHITECTURE

The problems for the symbol systems approach to cognitive architecture described in the previous section derive directly from the assumption that the structural architecture of the cognitive system has essentially the same form as the structural architecture of the digital computer. The interactive approach to cognitive architecture resolves the problems by giving up this assumption and replacing it with a view of the structural architecture of the cognitive system that is consonant with Turing's original analysis and with the approach of theorists such as Clark and Hutchins, who see agent-environment interactions as crucial.
From the earlier discussion of Turing's work in Section 2, it is evident that the structural architecture of the Turing machine was derived in the first place from the example of a human computer working with paper and pencil. This example provides the basic conception of structural architecture to which I believe we should return. The import of this conception is, however, much broader than we tend to think. What Turing's formal analysis shows is that the interactions between two finite sets of physical entities Q and S can be considered to constitute a computation when they meet certain requirements and can be understood in terms of a finite state transition function \( \phi \) of the kind discussed in Section 2. The requirements include properties like finiteness, discreteness, and permanence. Turing understood the set Q as a set of "states of mind" and S as a set of symbols of the kind used for arithmetic, but the formal analysis does not preclude other realizations. In particular, it does not preclude understanding S as including entities other than alphanumeric symbols.

The interactive approach to cognitive architecture described here is based on just such a broad conception of Turing's analysis, which takes the structural distinction between control and external medium as fundamental. Under this conception, the brain instantiates not the full task architecture of a Turing machine, but only the control architecture. The input architecture is found in the external environment. This leads to a view of cognitive computations as processes of structured interaction between the control architecture of the brain and the input architecture of the environment. Cognitive computation, in other words, is irreducibly world involving; the cognizer is embedded in an environment which constitutes part of the cognitive architecture. One important aspect of this approach is derived from the fact that the control architecture and the input architecture of a Turing machine are co-designed. Thinking about human cognitive architecture as a TM architecture that involves both the brain and the external environment suggests that brain and environment are also, in some sense, co-designed. Pursuing this idea fosters a way to study cognitive architecture in which interaction between brain and environment is a central topic, not a peripheral concern to be added on after "central" cognition has been understood.

The interactive approach solves the transduction problem, in principle, by not hypothesizing internal symbolic encodings of objects and events. This does not imply that internal states are not or cannot be representational. However, complete specificity of representations is not needed because entities in the world represent themselves. Stimuli are registered in ways which may have permanent representational effects, but those effects may, for example, alter the firing rate of a given neural ensemble rather than record specific aspects of the stimulus in symbolic form. This suggests that the function of a representation may have more to do with modifying the system's sensitivity to a class of inputs rather than storing a detailed inventory of its features. There is a similarity in this regard to the work of Maturana & Varela (1980), although their approach leads to a relativistic epistemology which is not supported here.

The interactive approach also resolves the evolutionary problem, in principle, because the internal architecture is not hypothesized to be general purpose. As the control architecture of a TM, it is special purpose with respect to its input environment, and thus evolvable in principle. This does not mean that humans cannot behave as universal computers. What
it does mean is that the locus of programs is moved from the internal architecture out into
the environment. This issue is discussed further in the State/Symbol Interaction section.

Turing's Distinction Between Finite Mind and Infinite Tape

Before developing the interactive proposal in more detail, I provide further grounding for
it in the idea of an abstract computing machine. As has been made clear, Turing conceptual-
ized the finite state control machine as a model of the mind of a human "computer" and
the tape as a model of the paper on which a human would work through a calculation. Thus
the tape was not part of the internal cognitive architecture of the cognizer. Moreover, the
distinction between the two parts of the TM was based on what Turing saw as fundamental
differences between the mind and the external medium represented by the tape. The central
topic of the 1936 paper was what Turing called the "computable" numbers, which he
defined as the real numbers "whose decimals are calculable by finite means." The key
question is why there should be the restriction to finite means. Part of the answer lies in
Turing's use of the notion of a computing machine to show that the Hilbertian Entschei-
dungsproblem was unsolvable. This application required Turing to formalize the notion
of a definite method for deciding a mathematical assertion, and the computing machine
was his chosen formalization. A computing machine whose description was infinite would
clearly be unsatisfactory for this purpose. However, the application to the Entscheidungs-
problem appears not to be the whole answer to the question of finite means. At the start of
the paper, Turing briefly justified the restriction to finite means by reference to "the fact
that the human memory is necessarily limited" (Turing, 1936-7, p. 117). This reference to
the limitation of human memory is independent of the application of the machine concept
to the Entscheidungsproblem. It would have sufficed, for that purpose, to have specified
that a computing machine must be finite purely on the grounds of the need for definiteness.
The reference to human memory is not needed in the context of the Entscheidungsproblem
and seems to represent an additional, independent, line of thought. This suggestion is borne
out by Turing's further discussion of the issue.

The justification which he advanced in his further discussion is characteristically terse.
He said, "If we admitted an infinity of states of mind, some of them will be 'arbitrarily
close' and will be confused." Turing (1936-7, p. 136). This hardly counts as an argument,
but seems to amount to a commitment to what would now be called the supervenience of
states of mind on states of the brain. (cf. Kim, 1993). Very roughly, if states of mind superven-
ve on states of the brain then two states of mind \( q_i \) and \( q_j \) cannot be distinct unless there
are distinct brain states to which the states of mind stand in an appropriate relation. Another
way to put this is to say that there cannot be more functional states of mind than there are
distinct information bearing states of the brain. Since there can be only a finite number of
distinct information bearing brain states there can also be only a finite number of functional
states of mind supervening on them. Turing's idea that states of mind will become arbi-
trarily close and confused if an infinity of them is allowed must rest on something like the
concept of supervenience since there is obviously no difficulty in defining an infinite num-
ber of functional states, \( q_1, q_2, \ldots \) if physical instantiation is not an issue. Turing's biogra-
pher suggests that by the time the work on computable numbers was written, Turing was
becoming "a forceful exponent of the materialist view." Hodges (1983, p. 108). This adds weight to the suggestion that the finite restriction on the number of states of mind to avoid "confusion" was a recognition of their supervenience on a finite substrate.

The situation with respect to the tape is entirely different. Its capacity has to be infinite. The requirement was implicit but entirely clear in Turing's distinction between "circular" and "circle-free" machines in the 1936 paper and was made explicit in later work. In 1947, he said of his earlier work "I considered a type of machine which had a central mechanism, and an infinite memory which was contained on an infinite tape...It was essential in these theoretical arguments that the memory should be infinite." Turing (1947, pp. 106–7). Quite clearly, if human memory is necessarily limited by its supervenience on a finite substrate and if tape memory is necessarily infinite then we cannot suppose that Turing thought of the tape of a Turing machine as instantiated in the same medium as the control. Hence the structural distinction between them.

Moreover, the distinction between control and tape is not just quantitative. The control mechanism is the active part of the system and the tape is the passive part. It is perfectly true that the behavior of a Turing machine is jointly determined by the current internal state and the symbol scanned. But this does not imply that the tape is an active entity in the way in which the control is, any more than the fact that signposts to some extent determine which way I turn when I want to drive to Banbury rather than Bristol implies that they are active entities. Turing did not himself draw particular attention to the qualitative differences between control and tape, which he presumably took to be entirely obvious, but von Neumann was quite explicit about the distinction. Describing the way in which a Turing automaton modifies its tape he said, "This tape is not itself an object which has states between which it can move of its own accord. Furthermore, it is not finite, but is assumed to be infinite in both directions. Thus this tape is qualitatively completely different from the automaton" (von Neumann, 1987, p. 478). It seems, therefore, that the interactive interpretation restores the original, principled, idea that the Turing machine models the interaction of the cognitive system with qualitatively different external resources.

The structural architecture of agent and environment proposed under the interactive conception of the Turing machine construes the mental states of an agent as the control architecture of a TM, i.e. as the set Q of its internal functional states. These states interact with a set S of symbols located in the external environment whose nature and sequencing constitute the input architecture of the TM. The interactions proceed in accordance with a state transition function $\phi$ such that $\phi(q_i, s_j) = (s_r, q_x, d_l)$. I now consider this schema in more detail.

**The Interactive Interpretation of Internal States**

The first point to note is that the fixed set of internal states $Q$ is hypothesized to have both representational and control functions and to support perception, thought and action simultaneously. This follows from the structural assumption that the task architecture of the cognitive system is distributed between the agent and the environment. Symbol structures, the components of the input architecture, are located in the environment and thus, by hypoth-
esis, cannot play the representational role in the interactive approach that they do in the internalist approach.

One objection to this idea is that the internal states of Turing machines are global and indivisible and hence cannot have the kind of structure which the interactive approach requires for states to have both representational and control components. I think this objection confuses the way in which states are described and the way in which they might be implemented. The description of a state as global relates to the fact that it is treated as a functional unit and does not imply that the state must be structurally atomic when implemented. It might, for example, be implemented as a particular arrangement of a set of parts. Take a simple gearbox with one forward gear “F” and one reverse gear “R”. F and R are global states of the gearbox but this has no implications for the implementation structure. I take it, therefore, that the idea of global states which have structured implementations is sound in principle.

It is apparent, however, that construing mental states as the elements of the set \( Q \) of the control architecture of a Turing machine, even allowing for structural complexity in their implementations, tells us almost nothing about them, other than that they are discrete and that there are only finitely many of them. The principal reason for this paucity of information is that Turing was engaged in an essentially reductive enterprise. He wanted to pare away the “inessential” aspects of mental states to arrive at the functional core. What needs to be borne in mind is that the construal of what is essential and what is inessential is very much relative to the goal of a particular analysis and cannot be taken as absolute. The goal of Turing’s analysis was to give an account of the essential processes involved in the effective computation of a real number by the application of specified rules. From the point of view of a theorist with such a goal, most of the activities in which most of us engage when calculating with paper and pencil are irrelevant. That we get bored, make mistakes, become distracted and tired, or doodle on the edge of the paper are all irrelevant to the theorist whose goal is to understand the processes that are involved in the successful completion of a task in numerical computation. The fact that numerical computation is just one of a number of processes which are executed in parallel is inessential to its analysis. However, if one’s aim is to understand the nature of the states of a cognitive architecture within which a multiplicity of processes are taking place in parallel, a more elaborated notion of mental state is needed which integrates, among others, affective and conative as well as cognitive processes. Since Turing’s pioneering work on serial systems, a variety of formal approaches to parallel computation have been developed, any of which might serve as the basis for such an analysis of mental states. The approach of Hoare (1985), for example, provides both a process algebra and proposals for the implementation of processes.

A further problem for the interactive proposal about the nature of internal states is that it seems to bind both the content and process of thinking too closely to the immediate stimulus environment. The problem arises because of the form of the state transition function. When \( S \) is taken to be a set of entities in the external world, the fact that \( s_i \in S \) is an argument to the transition function appears to bind the action of the machine to the immediate stimulus environment. That being so, one might ask for example, how it is possible for me as I sit at my desk in London, to imagine myself walking in the mountains in Scotland. The
answer that the symbol systems theorist can give to questions of this sort is based on the idea that thinking can access a stored, internal representation, by hypothesis in the form of a symbol structure, which can then be processed internally, independently of the current input environment. This solution is unavailable to the proponent of the interactive interpretation because internal symbol structures are not postulated as the locus of representations and because state transitions are, in part, a function of the immediate input.

The problem of independence of processing is resolved in principle by demonstrating that the formal apparatus provides the means to escape undue stimulus determination. Consider the following two transition schemas: a) \( \Phi(q_p s_j) = (s_p q_p d_i) \); b) \( \Phi(q_p s_j) = (s_b q_p d_j) \). Schema a describes those cases where the transition function changes the internal state but does not change the output symbol. This schema allows for arbitrarily long transition sequences through different internal states under constant input and output conditions. This is most clearly seen when \( d = N \), i.e. when the input location remains the same, but does not depend on this auxiliary condition. What this schema shows is that a constant external environment is compatible with changing internal activity. The schema might model situations such as those where I am sitting at my desk, looking at the bookshelf in front of me and thinking. The input conditions I experience are (approximately) constant but my thoughts are free to vary. Schema b describes those cases where the transition function changes the output symbol but not the internal state. What this schema shows is that a changing external environment is compatible with an unchanging internal state. This schema might model situations like expecting the telephone to ring, which commonly endure through changing environmental inputs. The conclusion to draw from these two transition schemas is that the requirement for a proximal input as an argument to the transition function does not rule out the independence of thinking under the interactive interpretation.

The content problem is addressed by adverting to the structured nature of internal functional states. It might be, for example, that certain aspects of an internal state are primarily dedicated to control and others to representation. This need not imply that the representational aspects of internal states are symbolic in the sense described by symbol systems theorists. For example, van Gelder (1990) and Clark (1993) both discuss ways in which structured representations can be achieved other than through symbol systems. It is clear that memory and control functionality are often intertwined and thus the theoretical link between them which the interactive interpretation of the TM proposes has some prima facie plausibility. The proverbial notion that the child who is burned fears the fire describes the situation. The memory of the pain felt on a previous occasion serves to initiate the appropriate actions to keep at a safe distance in the future.

Damasio (1989), (cf. also Cohen & Eichenbaum, 1993), proposes an organization of the substrates of recognition and recall that rests on a substantial body of neuroanatomical and neurophysiological evidence and is compatible with the functions assigned to internal states by the interactive interpretation of the TM. Damasio rejects the idea that perception depends on symbolic representations being transformed by a unidirectional cascade of processors that provide successively more concatenated analyses of initially distributed input features, leading finally to single site multimodal cortices in which fully integrated repre-
sentation is achieved. Such a view implies that representational feature tokens are passed from the sensory periphery to higher associative processes, which combine and manipulate them to produce meaningful percepts. Damasio argues that token passing of this kind does not occur. "No representations of reality as we experience it are ever transferred in the system; that is, no concrete contents and no psychological information move about in the system." Damasio (1989, p. 57). He proposes instead that representational feature tokens, which he calls "the fragment record," remain in neural ensembles at or near the sensory portals and motor output sites. Integration of these feature tokens is achieved by time-locked retro-activation of the feature tokens via feedback projections from "convergence zones." Convergence zones are neural ensembles that receive projections from multiple sites in both sensory and motor primary and early association cortices. Input to them is thus many to one. Reciprocal, one to many, feedback from convergence zones to feature token records reactivates specific combinations of such assemblies. These reactivated combinations also form the basis for both recognition and recall.

The system proposed is hierarchical. Local, or first order, convergence zones which receive multiple, unimodal inputs, themselves project to other local convergence zones and to higher order, multimodal convergence zones and receive feedback from them. The patterns of feedforward and feedback connectivity between local and higher order convergence zones and their functional relationships are hypothesized to be the same as those between fragment records and local convergence zones. Higher order convergence zones thus serve to reactivate combinations of local convergence zones.

A striking feature of Damasio's proposal is that content is positionally defined. It is the site of a fragment record which determines what it is about, rather than, an origin-independent code. The sites of convergence zones also define their content. The neural geography of convergence zones thus provides the basis for psychological distinctions among domains. Convergence zones are not themselves representational but serve to reconstruct representations.

Two features of Damasio's proposal are of particular interest to the interactive interpretation of the TM. The first is the hypothesis that reciprocally connected sets of structures comprising fragment records and convergence zones subserves memory, perception, and motor control. The degree of integration implied by this proposal suggests that it is entirely reasonable to think of the global functional states of the system as supporting memory and control simultaneously and thus establishing the bi-directionality of thinking and perception, which the interactive interpretation suggests. Second, the suggestion that neural computation does not involve the transmission and manipulation of internal symbol structures supports the interactive approach to representation. This latter point suggests that although there are representational "atoms" in the cognitive system, that is the fragment records, there are no atomic symbols that can be combined into arbitrary representational structures in the manner of symbolic primitives in computers. We note, therefore, a key similarity between Damasio's neurological theory and the interactive interpretation of the TM. Damasio's proposal is sufficiently explicit that it may be possible to model his ideas in a process algebra like that of Hoare (1985).
External Symbols and Symbol Types

The second notable characteristic of the interactive interpretation of the TM is that it extends cognitive architecture into the environment by interpreting external symbol structures as part of the structural architecture. One apparent limitation of this proposal is that Turing machines deal only with discrete symbols whereas minds also deal with objects, events, and other people. The starting point for an answer to this objection is that it is the properties of type identity and discriminability which are crucial, not the specific forms that conventional symbols take. It is argued that type identity can be predicated of a wide variety of entities. The key idea is to broaden the notion of a symbol to include what I shall call "natural" symbol types as well as conventional types.

Consider the minimal symbolic repertoire for a Turing machine which is a set with two elements e.g. \{0,1\}. The critical theoretical property of the elements is type identity (cf. Pylyshyn, 1984, pp. 50–1). Any one token of a given symbol is functionally indistinguishable from any other and distinct from the tokens of any other symbol regardless of context. The basic symbol types are also simple. They are not constructed out of more primitive elements. A small repertoire of primitives is sufficient because compound symbols can be formed by concatenating tokens of the primitives as in “01” for example. The individual symbol tokens have no intrinsic semantics, nor do compounds constructed from them. The processing of a Turing machine operating with conventional symbols is purely syntactic. Interpretations are assigned by the designer of a particular TM or the engineer who designs a processor and so forth. The mapping from 1,11,111,... to the numerals 1,2,3,... in the Turing machine of Table 1 is an example. Cognitive theorists acknowledge that there is a difficulty in understanding how the symbols in the presumptive internal alphabet of the human cognitive system acquire their interpretations in the absence of a designer or engineer. Harnad (1990) has called the difficulty the symbol grounding problem.

The interactive interpretation of the TM formalism offers a perspective on the symbol grounding problem based on the distinction between conventional symbols and natural symbols. The point is this: If it can be shown that there are natural symbols whose interpretations are intrinsic and that conventional symbols derive their semantic@ from natural symbolization, then the symbol grounding problem can be resolved.

To see how natural symbols might arise consider the fear of snakes as an example. Buss (1995) lists a fearful reaction to snakes as one type of response which might plausibly have had a selective advantage even though not all species are venomous. Morphologically and in their method of locomotion, snakes are all rather similar and distinct from other classes of reptiles even though there is considerable variation in the characteristics of particular species. It is not hard to see how evolution might have selected for a generalized avoidance/fear reaction to snakes based on the “natural syntax” of their shape and movement which type-identified them as creatures to which a particular class of response is appropriate. For example, Cheney and Seyfarth (1992) describe how East African vervet monkeys (Erycopsithecus aethiops) have a distinct alarm call which they use to signal the presence of the python (Python sebae). This call differs both acoustically and in terms of the behavior it elicits from the calls that vervets use to signal other predators. Thus we arrive at the idea
of a natural symbol as a type which has a built-in grounding arising from its adaptive significance. The type identification would of course hardly be perfect. There would be a penumbra of misidentifications, both false positives and negatives, but an inheritable selective advantage deriving from successful avoidance could build in an automatic response, which would be tuned by the core cases of genuine snake identification.

As another example, consider the existence of marked discontinuities in the physical environment, cliff edges for example. These discontinuities have a significance for land mammals which they do not have for birds or fish for example. They are places where you can fall off and be killed, and they are places where you can gain an extensive view of other parts of the environment. They are significant features which foster both approach and avoidance. Falling off a cliff edge has such a detrimental effect on fitness that it is easy to see that a mechanism promoting caution would be adaptive. The famous visual cliff experiment of Gibson & Walk (1960) might constitute evidence for just such a mechanism. Marked discontinuities can be type-identified as such without difficulty and appear to have an intrinsic meaning built-in. More generally, the idea of natural symbols implies a built-in, evolved relationship between the architecture of the individual and the architecture of the environment (cf., Orians & Heerwagen, 1992; Kaplan, 1992). The development of conventional symbols from such natural bases needs to be clarified, but a variety of hypotheses can be found in the literature. Mithen (1996), for example, locates the origins of conventional symbolism in the de-modularization of specialized intelligences which led to what he calls "cognitive fluidity"; Donald (1991) suggests that symbolism originated in a number of phenomena including a shift in the relative importance of auditory and visual representation and the development of analytic thought.

State/Symbol Interaction

The third, illuminating aspect of the interactive approach to the TM formalism is the insight it gives into the idea that the human mind and aspects of the environment are co-designed and interlocked. There are a number of points to consider. The first, very simple, point is that machines that make systematic use of an external environment are computationally more powerful than ones that do not. Turing machines, for example, are more powerful than finite automata because they make use of external tapes. The general nature of this increased power can be studied in the context of simple environments like the tape of a TM. More tapes and/or read/write heads do not further extend the computing power of a system. We can, therefore, think of the TM as the general basis for studying machine-environment interactions. From one point of view, the capacity to use external symbol systems for reading, writing, and calculating is perhaps the most characteristic and most powerful extension of the human intellect that has been developed. The paradigmatic model for this capacity is the universal machine, which interprets and acts upon sets of instructions. But the universal machine depends on the basic interlocking of state and symbol and the unfolding of behavior through sequences of state transitions, which is characteristic of Turing machines generally. The central point about the use of an external medium is that the results of earlier actions change the medium in ways that can affect later behavior. Thus the environment provides both data storage and feedback. This type of interaction can lead to
remarkably complex behavior even in very simple machines, some of which have been
documented by Machlin and Stout (1990). One five-state Turing machine which they
describe holds a number of lessons for theorists of cognitive architecture. It is a machine
that was discovered by J. Buntrock and H. Marxen in 1989. The machine uses just two
symbols \{#, l\} and is started in state \(q_1\) on a blank tape. Its machine table is shown in
Table 2.

One index of the complexity of behavior of Turing machines which eventually halt is
the number of steps that they perform. One might intuitively expect that a five-state
machine, with an alphabet of two symbols started on a blank tape, would halt rather quickly
if it were going to do so at all. However, Buntrock and Marxen’s machine executes
23,554,764 steps before halting. This remarkable value is not predictable from an exami-
nation of the internal structure of the machine. It is an emergent property of the interaction
between the machine and the sequence of symbols that it leaves on its tape.

Machines with more states and larger alphabets might reasonably be expected to display
even more complex behavioral dynamics. If the structural architecture of the cognitive sys-
tem extends into the environment in the way that the interactive approach to the TM sug-
gests, Buntrock & Marxen’s machine shows that behavior cannot be understood through an
analysis of either the internal structure of the cognitive system or the structure of the envi-
ronment alone. There can be no substitute for studying behavior interactively, as it unfolds.
Another interesting facet of the machine is the light it throws on the nature of computation
generally. We are accustomed to thinking of computation as rational, goal-directed activ-
ity, primarily because computers are normally used in the pursuit of rationally specified
goals. Buntrock and Marxen’s machine shows that such a view of computation is some-
what restricted. There is no doubt that their machine executes a computation, but its behav-
ior defies rational analysis. There is no obvious interpretation that maps its behavior onto a
task in a fashion analogous to the interpretation that maps the behavior of the TM of Table
1 onto the task of multiplication. The exotic behavior of machines of this kind may perhaps
provide a partial answer to the criticism that computational thinking in cognitive science is
unduly dominated by rationality and goal directedness.

Humans are, nevertheless, often goal directed and do sometimes act rationally. More-
over, they are also capable of understanding and acting on instructions. It is with respect to
this capacity that we should think about the way to construe computational universality in terms of the interactive interpretation of the TM. The task architecture of a universal machine can be realized in the interactive model of structural architecture in a straightforward and intuitively compelling way. The control architecture of the brain includes the mechanisms to interpret instructions, and the input architecture of the environment contains symbolic representations of task architectures to be simulated. Thus the fixed (modulo changes arising from learning, disease, ageing, etc.) control architecture of the human brain can instantiate a universal machine in conjunction with an input architecture which provides both data and instructions. When cooking using a recipe book, for example, the control architecture of the cook’s brain uses the input architecture of a recipe to control the interaction with the kitchen and ingredients. The cook does not have to internalize a recipe in order to use it. It suffices just to follow it. The behavioral flexibility of the system rests on the fact that both instruction types and their tokens are variable. The cook can use different recipes and can also follow instructions for other types of activity. The interactive approach to the TM thus construes computational universality not as demonstrating that the brain is programmable in the way in which a computer is by internal structural modification, but that the cognitive system is universal because the brain is capable of acting on an indefinitely large number of sets of external instructions. This view provides a cogent way of thinking about symbolic artifacts generally and the relation between cultures and individuals. Cultures can be thought of as extensive collections of programs, or sets of instructions, which contribute to the behavior of those who use them.

VII. CURRENT COGNITIVE THEORY AND THE INTERACTIVE APPROACH

The interactive approach to cognitive architecture described in this paper is relevant to a number of areas of activity in cognitive science. Three examples are discussed briefly; they are connectionism, the “wild” cognition studied by Edwin Hutchins and the computational approach to evolutionary psychology of Leda Cosmides and John Tooby.

Connectionism

The interactive interpretation of the Turing machine leads to a view of cognitive architecture which is incompatible with the architecture of symbol systems. It is interesting to consider briefly what the interactive view suggests about connectionism, the other main strand of computational theorizing in cognitive science.

A clear link can be established between connectionist networks and Turing machines via the work of McCulloch and Pitts (1943). The most powerful networks considered by McCulloch and Pitts were what they called nets with circles, that is, networks with feedback loops. Nets with circles compute the same functions as connectionist networks with feedback, (e.g. the recurrent networks of Elman, 1990). The difference is that recurrent connectionist nets learn the function to be computed rather than having it specified. McCulloch and Pitts claimed that nets with circles, when equipped with scanners and a tape, could compute the same numbers as a Turing machine, whereas nets with circles but
without scanners and a tape were less powerful. Thus we find in McCulloch and Pitts, (1943) the same sort of separation of components as we find in Turing (1936-7).

Applying the logic of the interactive approach, and thinking of connectionist nets rather than McCulloch-Pitts nets, suggests that connectionism is a way of exploring the part of the cognitive architecture which is analogous to the finite state controller of a Turing machine. There are some differences of emphasis, perhaps most notably the fact that Turing described the finite state controllers of his machines in terms of discrete states, whereas the activation levels of connectionist networks are continuously variable. But Turing was thinking of functional states rather than of ways of implementing them. The idea that a finite state controller might be implemented in a continuous substrate is one that he would have found perfectly acceptable. It appears, therefore, that the connectionist and interactive approaches to cognitive architecture are theoretically compatible in that both see the organism as a finite controller embedded in a wider environment that forms part of the architecture. The distinction between discrete and continuous approaches is considered further in Section 8.

Cognition in the Wild

In recent publications, Hutchins (1995 a,b) has argued for an approach to cognitive computation that broadens the concept to include interactions with external artifacts and with other people: "...I want the sort of computation that cognition is to be as applicable to events that involve the interaction of humans with artifacts and with other humans as it is to events that are entirely internal to individual persons." Hutchins (1995a, p. 118). Hutchins seeks to understand cognition "in the wild" as he puts it, that is cognition as it occurs naturally in culturally constituted, everyday, human activity. He is particularly concerned to emphasize the importance of external artifacts as representational components of cognitive systems. Hutchins argues that tools do more than simply amplify the cognitive abilities of task performers. They transform the task to be performed by representing it in a different way. Navigation tools, for example, are repositories of knowledge, that is they are representational media that also provide constraints on patterns of action. In this sense they can be thought of as filling the same functional role as an external set of instructions for a universal machine interpreter. From the point of view of the interactive interpretation of Turing's analysis, there is no reason why a "program" should not exist in the form of an artifact, provided that the control architecture has the physical capacity to interact with it.

Hutchins' approach has a great deal in common with the interactive approach to the TM, but he diagnoses Turing's analysis of computation as one of the reasons why the cognitive science tradition has gone astray. He suggests that "For Turing, the essentials [of mathematical problem solving] evidently involve the patterns of manipulations of the symbols, but they expressly do not involve the psychological processes which the mathematician uses in order to accomplish the manipulations." Thus, he continues, when symbol processing is modelled in an automatic system, "The mathematician who was a person interacting with a material world is neither modeled by this system nor replaced in it by something else." Hutchins (1995a, pp. 362-3). As the discussion of Turing's analysis in this paper shows, this is not correct. Turing did model the psychological processes in the mind of the
mathematician as the control architecture of the Turing machine. What has happened in the
symbol processing tradition is not that the mind is not modeled, but that the relation
between it and the external world has been incorrectly construed. However, the conclu-
sions Hutchins reaches are in agreement with the interactive approach. What this paper
suggests is that Hutchins' approach is an example of how the mainstream of computational
thinking in cognitive science ought to have developed from Turing's insights, not an exam-
ple of how to replace Turing's analysis.

Evolutionary Psychology
The interactive interpretation of Turing's analysis also provides a natural model for a com-
putational approach to evolutionary psychology. Evolutionary psychology "unites modern
evolutionary biology with the cognitive revolution," Barkow, Cosmides, & Tooby (1992,
p. 3). A central postulate of the approach is the existence of a universal human nature func-
tioning at the level of evolved psychological mechanisms. These mechanisms are hypo-
thesized to be adaptations, which are domain-specific, contentful, special-purpose
mechanisms constructed by natural selection over evolutionary time to solve the recurrent
adaptive problems experienced by our hominid ancestors. In terms of the interactive
approach described here, these mechanisms may be thought of as the control architecture
of an evolved TM.

Tooby and Cosmides think of evolutionary psychology as a computational approach to the
study of the mind because they propose that adaptations are computational mechanisms
which should be described and analyzed in functional terms: "For the purpose of discover-
ing, analyzing, and describing the functional organization of our evolved psychological
architecture, we propose that the information-processing language of cognitive science is
the most useful." Tooby and Cosmides (1992, pp. 63–4). This proposal is based on two
claims: first that cognitive descriptions are best suited to the task of describing the invariant
functional organization of complex psychological adaptations, and second that natural
selection operates primarily on this functional organization. However, evolutionary psy-
chologists also claim that, although the mind is a computational system, its base mecha-
nisms do not constitute a general-purpose, universal computer as the symbol systems
approach proposes, but rather "an intricate network of functionally dedicated computers,
each activated by different classes of content or problem, with some more general-purpose
computers embedded in the architecture" Tooby and Cosmides (1992, p. 94). Thus cogni-
tive architecture is essentially a collection of special-purpose adaptations. A particularly
important notion is the intimacy of the connection between an adaptation and those recur-
rent features of the environment with respect to which it was selected: "Adaptations evolve
so that they mesh with the recurring structural features of the environment in such a way
that reproduction is promoted in the organism or its kin. Like a key in a lock, adaptations
and particular features of the world fit together tightly, to promote functional ends." Tooby
& Cosmides (1992, p. 69). The interactive interpretation of Turing's analysis provides a
natural foundation for this kind of thinking. Furthermore, it provides a formal framework
within which to address the difficult problem of the way in which adaptations interact. Mil-
ner (1989) and Hoare (1985) provide calculi which address the problems of understanding
concurrent and communicating processes and might provide the basis for a formal study of communications among aggregates of adaptations.

CONCLUSIONS

This paper presents an interpretation of Turing's analysis of computation that is faithful to his original conception and provides a good fit to the current trend towards interactive agent-environment styles of explanation of cognition and cognitive processes. In both these respects, this interpretation marks a departure from the traditional symbol systems approach, which the arguments of Section 5 suggest is untenable. There have been suggestions in the recent literature that computational thinking of any kind is no longer required in cognitive science. Thelen and Smith (1994, p. xix) for example, "categorically reject machine analogies of cognition" in favor of "a vocabulary suited to a fluid, organic system, with certain thermodynamic properties." They suggest that the theory of continuous dynamic systems provides that vocabulary. By reconceptualizing Turing's analysis, I think it can be seen that Thelen and Smith are proposing a false dichotomy. The interactive approach to cognitive computation and dynamical systems theory have complementary aims and need not be opposed. The difference is that one provides a continuous model and the other a discrete model. Beer has made the links explicit:

"[I]t should be noted that any system with finite state which evolves deterministically can be described using the concepts of dynamical systems...For example, the transition table of a finite state machine defines a flow on a discrete state space. The lack of a metric on this state space limits the dynamical behavior that a finite state machine can exhibit, but such concepts as initial state, trajectory, flow, attractor, equilibrium point, limit cycle, basin of attraction...still apply. Thus the present framework may still be useful even if the continuity assumption should turn out to be inappropriate." Beer (1995, p. 206).

And that, surely, is the issue that future research must address. To what extent do we need both continuous and discrete models to understand cognition? Some cognitive capacities, such as external symbol use, may require only discrete treatment; others, such as the capacity to structure immediate behavior flexibly and appropriately may require continuous treatment. It may be that Turing's model of internal states as discrete will turn out to be too simple for the understanding of anything other than the deterministic rule-governed activity for which they were first proposed. We may ultimately require an approach to cognitive architecture that combines elements of both discrete and continuous models.

Acknowledgments: I would like to thank my colleague Bradley Franks and two of our students, Simon Hampton and John Stringer, for discussion of a draft of the paper. I am also grateful to Axel Cleeremans and Marco Giunti for refereeing the paper, and to Bill Clancey for his insightful editorial suggestions.
NOTES

1. Page references to Turing's paper are for the reprint in (Davis, 1965).
2. The term "functional architecture" would have been ideal but could lead to confusion with Pylyshyn (1984) who uses it to describe something similar to what is called "structural architecture" here.
3. The terms "task" and "function" are used interchangeably.
4. Machlin and Stout (1990) give the figure as 23,554,768 steps. The result reported is from the author's own simulation of the machine.

REFERENCES


