Symbolic Artificial Intelligence and Numeric Artificial Neural Networks: Towards A Resolution of Dichotomy

TR94-14
Vasant Honavar

August 18, 1994

Iowa State University of Science and Technology
Department of Computer Science
226 Atanasoff
Ames, IA 50011

12

SYMBOLIC ARTIFICIAL INTELLIGENCE AND NUMERIC ARTIFICIAL NEURAL NETWORKS: TOWARDS A RESOLUTION OF THE DICHOTOMY

Vasant Honavar

Department of Computer Science
Iowa State University
Ames, Iowa 50011

1 INTRODUCTION

The attempt to understand intelligence entails building theories and models of brains and minds, both natural as well as artificial. From the earliest writings of India and Greece, this has been a central problem in philosophy. The advent of the digital computer in the 1950's made this a central concern of computer scientists as well (Turing, 1950). The parallel development of the theory of computation (by John von Neumann, Alan Turing, Emil Post, Alonzo Church, Steven Kleene, Markov and others) provided a new set of tools with which to approach this problem — through analysis, design, and evaluation of computers and programs that exhibit aspects of intelligent behavior — such as the ability to recognize and classify patterns; to reason from premises to logical conclusions; and to learn from experience.

In their pursuit of artificial intelligence and mind/brain modelling, some wrote programs that they executed on serial stored-program computers (e.g., Newell, Shaw and Simon, 1963; Feigenbaum, 1963); Others had more parallel, brain-like networks of processors (reminiscent of today's connectionist networks) in mind and wrote more or less precise specifications of what such a realization of their programs might look like (e.g., Rashevsky, 1960; McCulloch and Pitts, 1943; Selfridge and Neisser, 1963; Uhr and Vossler, 1963); and a few took the middle ground (Uhr, 1973; Holland, 1975; Minsky, 1963; Arbib, 1972; Grossberg, 1982; Klir, 1985).

It is often suggested that two major approaches have emerged — symbolic artificial intelligence (SAI) and (numeric) artificial neural networks (NANN or connectionist networks) and some (Norman, 1986; Schneider, 1987) have even suggested that they are fundamentally and perhaps irreconcilably different. Indeed it is this apparent dichotomy between the two apparently disparate approaches to modelling cognition and engineering intelligent systems that is responsible for the current interest in computational architectures for integrating neural and symbolic processes. This topic is the focus of several recent books (Honavar and Uhr, 1994a; Goonatilake and Khebbal, 1994; Levine and Aparicioiv, 1994; Sun and Bookman, 1994). This raises some important questions: What exactly are symbolic processes? What do they have to do with SAI? What exactly are neural processes? What do they have to do with NANN? What (if anything) do SAI and NANN have in common? How (if at all) do they differ? What exactly are computational architectures? Do SAI and NANN paradigms need to be integrated? Assuming that the answer to the last question is yes, what are some possible ways one can go about designing computational architectures for this task? This chapter is an attempt to *explore* some of these fundamental questions in some detail.

This chapter argues that the dichotomy between SAI and NANN is more perceived than real. So our problems lie first in dispelling misinformed and wrong notions, and second (perhaps more difficult) in developing systems that take advantage of both paradigms to build useful theories and models of minds/brains on the one hand, and robust, versatile and adaptive intelligent systems on the other. The first of these problems is best addressed by a critical examination of the popular conceptions of SAI and NANN systems along with their philosophical and theoretical foundations as well as their practical implementations; and the second by a judicious theoretical and experimental exploration of the rich and interesting space of designs for intelligent systems that integrate concepts, constructs, techniques and technologies drawn from not only SAI (Ginsberg, 1993; Winston, 1992) and NANN (McClelland and Rumelhart, 1986; Kung, 1993; Haykin, 1994; Zeidenberg, 1989), but also other related paradigms such as statistical and syntactic pattern recognition (Duda and Hart, 1973; Fukunaga, 1990; Fu, 1982; Miclet, 1986)), control theory (Narendra and Annaswamy, 1989) systems theory (Klir, 1969), genetic algorithms (Holland, 1975; Goldberg, 1989; Michalewicz, 1992) and evolutionary programming (Koza, 1992). Exploration of such designs should cover a broad range of problems in perception, knowledge representation and inference, robotics, language, and learning, and ultimately, integrated systems that display what might be considered human-like general intelligence.

2 SHARED FOUNDATIONS OF SAI AND NANN

This section makes clear that the fundamental philosophical assumptions and scientific hypotheses that have shaped both SAI and NANN research are identical. The shared foundations of SAI and NANN guarantee that there can be no fundamental incompatibility between the two paradigms for engineering intelligent systems or for modelling minds/brains.

2.1 SAI and NANN Share the Same Working Hypotheses

The fundamental working hypothesis that has guided most of the research in artificial intelligence as well as the information-processing school of psychology is rather simply stated: *Cognition, or thought processes can, at some level, be modelled by computation.* The philosophical roots of this hypothesis can be traced at least as far back as the attempts of Helmholtz, Leibnitz and Boole to explain thought processes in terms of mechanical (or in modern terms, algorithmic or computational) processes. This has led to the *functional* view of intelligence which is shared explicitly or implicitly by almost all of the work in SAI as well as NANN. Newell's *physical symbol system hypothesis* (Newell, 1980), Fodor's *language of thought* (Fodor, 1976), Minsky's *society of mind* (Minsky, 1986), Holland's *classifier systems* (Holland, 1986), and most neural network models (McClelland and Rumelhart, 1986; Kung, 1993; Haykin, 1994; Zeidenberg, 1989) are all specific examples of this functional view. In this view, intelligence can be characterized abstractly as a functional capability independent of any commitments as to the specific physical substrates that support the functions in question.

The primary means of describing the behavior of intelligent systems (be they natural or artificial) within the SAI paradigm is in terms of their having *knowledge* and behaving in light of that knowledge. This is the so-called *knowledge-level* description (Newell, 1990). But it is important to remember that descriptions at the knowledge-level represent just one of the many alternatives available. The choice of what description to use in modelling intelligence, as in science in general, must be based on pragmatic considerations as determined by aspects of the phenomena being modelled and the sorts of explanations being sought. Satisfactory accounts of system behavior often make use of multiple levels of description along with the necessary means of mapping descriptions at one level into descriptions at adjacent levels.

Perhaps not so obvious is the fact that exactly the same functional view of intelligence is at the heart of current approaches to mimic intelligent behavior within the NANN paradigm, as well as the attempts to understand brain function using the techniques of computational neuroscience and neural modelling. The earliest work on neural networks by Rashevsky (1960), McCulloch and Pitts (1943) and Rosenblatt (1962)nociteros62 — from which many of today's NANN models are derived — illustrates this point rather well. So does the emphasis on *computational* models in the recent book on this topic by Patricia Churchland and Terrence Sejnowski (1992) suggestively titled *The Computational Brain* (This is not to suggest that brain modelling can ignore the particular biological substrates that realize the computations in question but just that the computational characterization of what the brains do provides a useful class of explanations and predictions of mental phenomena). It is important to note that NANN models or theories of intelligence are stated in terms of abstract computational mechanisms just as their SAI counterparts. The differences (if any) from SAI are primarily in terms of the (often unstated) preference for functional descriptions of intelligent systems at a different level of detail using a different set of primitives.

Some have (somewhat misleadingly) used the term *neural level* to refer to such descriptions. Today's NANN models are almost certainly grossly oversimplified caricatures of biological brains (Shepherd, 1989; 1990). It is far from clear that NANN are more suited to modelling brains any more than SAI; Descriptions in terms of rules, tokens, and automata (typically associated with SAI systems) offer extremely useful descriptions of biological neural circuits at the cellular and molecular levels (Cooper, 1990). (More on this later).

It should be clear from the discussion above that both SAI and NANN paradigms essentially offer two different *description languages* for describing systems in general and intelligent systems — minds/brains (be they natural or artificial) — in particular. As pointed out by Chandrasekaran and Josephson (1994), the commitment of most SAI researchers to biology in describing intelligence does not typically go beyond the knowledge level. Although perhaps not as obvious is the fact that an analogous situation holds for NANN models. NANN researchers pick out some interesting or relevant aspects of biological phenomena, and then proceed to formulate an abstract functional model (using abstract models of neurons) for the selected aspects of the phenomena chosen. The abstract descriptions in both cases are usually stated in sufficiently general languages. One thing we know for certain is that such languages are all equivalent (see below). This provides absolute assurance that particular physical implementations of systems exhibiting mind/brain-like behavior can be described using the language of SAI or NANN irrespective of the physical medium (biological or silicon or some other) that is used in such an implementation. And the choice of the language should (as it usually is, in science in general) be dictated by pragmatic considerations.

2.2 SAI and NANN Rely on Equivalent Models of Computation

Turing was among the first to formalize the common-sense notion of computation in terms of execution of what he called an *effective procedure* or an algorithm. In the process, he invented a hypothetical computer — the *Turing machine*. The behavior of the Turing machine is governed by an algorithm which is realized in terms of a *program* or a finite sequence of instructions. Turing also showed that there exists a *universal* Turing machine (essentially a general purpose stored program computer with potentially infinite memory) — one that can *compute* anything that any other Turing machine could possibly compute — given the necessary program as well as the data and a means for interpreting its programs. Several alternative models of computation were developed around the same time including *lambda calculus* of Church and Rosser, *Post productions*, *Markov algorithms*, *Petri nets*, and *McCulloch-Pitts neural networks*. However, all of these models (given potentially infinite memory) were proved exactly equivalent to the Turing Machine. That is, any computation that can be described by a finite program can be programmed in any general purpose language or on any Turing-equivalent computer (Cohen, 1986). (However, a program for the same computation may be much more compact when written in one language than in some other; or it may execute much faster on one computer than some other). But the provable equivalence of all general purpose computers and languages assures us that *any* computation — be it numeric or symbolic — can be realized, in principle, by both SAI as well as NANN systems.

Given the reliance of both SAI and NANN on equivalent formal models of computation, the questions of interest have to do with the identification of particular subsets of Turing-computable functions that model various aspects of intelligent behavior given the various design and performance constraints imposed by the physical implementation media at our disposal.

3 KNOWLEDGE REPRESENTATION REVISITED

Knowledge representation and inference are perhaps among the most central research issues in the integration of SAI and NANN paradigms for modelling cognitive phenomena and engineering intelligent systems. This is evident from the fact that almost all the recent books on the integration of SAI and NANN paradigms (Honavar and Uhr, 1994a; Levine and Aparicioiv, 1994; Sun and Bookman, 1994) have devoted several chapters to this topic. It is therefore worth clarifying some basic issues about knowledge representation.

It is generally accepted in artificial intelligence and cognitive science that knowledge has to be *represented* in some form in order for it to be used. This is free of any commitment as to how a particular piece of knowledge is internally represented. However, implicit in this view is a commitment to use *some* language (e.g., first order logic, production rules, lambda calculus or LISP) to express and manipulate knowledge. Expressions in any such language can be syntactically transformed into any other sufficiently expressive language — this follows from the universality of the Turing framework. This is tantamount to saying that systems that use knowledge are simultaneously describable at multiple levels of description. And systems (such as living brains or robots) that exist in the physical world would have physical descriptions — just as the behavior of a computer can be described at an abstract level in terms of data structures and programs, or in terms of machine language operations that are carried out (thereby making the function of the hardware more transparent) or in terms of the laws of physics that describe the behavior of the physical medium which is used to construct the hardware.

Note that this view is entirely consistent with that of Churchland and others (Churchland, 1986; Churchland and Sejnowski, 1992) who have advocated the search for explanations of cognition at multiple levels. It is also important to not lose sight of the fact that such a system is embedded within an external environment with which it interacts in a closed loop fashion through sensors and effectors and its body of knowledge is *about its* environment, *its* goals, *its* actions. This is the essence of *grounding* (see below).

3.1 Nature of Knowledge Representation

Given the central role played by knowledge representation in functional approaches to understanding and engineering intelligence, the *nature of represen-*

tation is among one the most fundamental questions in artificial intelligence and cognitive science. Some insight into this question can be obtained by considering a concrete example. A common way to represent knowledge (at least in SAI) is with logic (Genesereth and Nilsson, 1987). It is worth emphasizing that logic is not the knowledge itself; it is simply a way of representing knowledge. (However, logic can be viewed as a form of meta-level knowledge about how to represent and reason with knowledge.) What logic enables us to do is represent the knowledge possessed by an agent using a finite set of logical expressions plus a process (namely, the inference rules of logic) for generating a (potentially unlimited) set of other logical expressions that are part of the agent's knowledge. Thus if we represented an agent's knowledge in the form of expressions a and b, and if $a \wedge b \models c$, the agent has (implicit) knowledge of c even though c was not part of the (explicit) representation. In fact, first order logic is universal in that it is powerful enough to represent essentially any knowledge that can be captured by a formal system. However, for certain types of knowledge to be used for certain purposes (e.g., knowledge of the sort that is captured by maps of some geographical region or a city), first order logic representation may be awkward, indirect, or overly verbose.

If on the other hand, we were to choose a different way of representing knowledge of an agent, one which did not permit any logical deduction, then the agent's knowledge could be limited to those expressions that were explicitly included in the representation. Such a representation is in essence, simply a lookup table for the expressions in question. Thus, (for lack of a better term), the *knowledge content* of a representation may be limited by restricting either the inferences allowed, the form of the expressions that may be included (that is, limiting the expressive power), or both. Indeed, it is often necessary to impose such limits on the power of representation in order to make their use computationally feasible (perhaps at the expense of logical soundness, completeness, or both).

In order for any system to serve the role of a representation (as used in most artificial intelligence and cognitive science theories) it must include: an encoding process that maps the physical state of the external environment into an internal state; processes that map transformations of the physical state of the external environment into appropriate (internal) transformations of the internal state; a decoding process that maps an internal state into a physical state of the external environment — all subject to the constraint that the result of decoding the result of application of internal transformations of an internal state (obtained from encoding a physical state of the environment) is the *same* as the result of directly transforming the physical state of the external environment. (This is perhaps a stronger requirement than is necessary — most likely influenced by the emphasis on logic. It is easy to see several ways of relaxing this constraint — by allowing the correspondence to be only *approximate* instead of exact, or attainable only with a certain probability. It must also be mentioned that not everyone agrees with this view of representation through encoding; See Bickhard, 1993 for a dissenting view). In short, representations are caricatures of selected aspects of an agent's environment that are operationally useful to the agent. Thus, certain mental operations on the representation can be used to predict the consequences of performing corresponding physical actions on the environment in which the agent operates.

Note that the internal transformations may be performed using LISP programs or production systems of SAI or by a suitably structured NANN. (Note however that the encoding and decoding processes are not purely symbolic because they have to deal with transduction or grounding. Also worth noting is the fact that most systems, be they SAI or NANN only simulate transduction and hence may lack grounding).

Newell (1990) proposes an additional requirement for representations — namely that the application of encoding (sensing), internal transformations, and decoding (acting) must be executable on demand to the extent required to serve the purposes of the organism (which could be viewed essentially as the sensed internal environment of needs, drives, and emotions).

3.2 Where Do The Representations Come From?

Representations may be discovered by organisms (or evolution) by identifying the right medium of encoders (transducers) and decoders (effectors) and the right dynamics for the transformations for specific tasks. This would lead to a large number of task-specific *analogical* representations. Indeed, strong evidence for such analogical representations are can be found in living brains: the retinotopic maps in the visual cortex and the somatotopic maps of the sensory-motor cortex provide good examples of analogical representations (Kandell and Schwartz, 1985).

Alternatively, or in addition, a set of encoders and decoders may be used in conjunction with the ability to compose whatever sequence of transformations that may be necessary to form a representation. Most SAI systems take this route to the design of representations — by using a sufficiently general language (e.g., LISP) that allows the composition of whatever functions that may be necessary to satisfy the appropriate representation laws. Most NANN systems take the same route as well — they just happen to use a different language with a different set of primitives for composing the necessary transformations.

Irrespective of the approach chosen, the discovery of adequately powerful, efficient, and robust representations for any non-trivial set of tasks is still a largely unsolved problem. This is where learning and evolutionary processes play a major role. They must build the representations that perception and cognition utilize. One of the most informative characterizations of learning to date is in terms of storage of results of inference in a form that is suitable for use in the future (Michalski, 1993). Learning can clearly provide an avenue for the discovery of the necessary compositions of transformations which is a major aspect of representation. However, note that both SAI and NANN systems presuppose the existence of some representation before they can discover other useful representations. (Therefore it appears that representations cannot come into existence without the existence of physical transducers and effectors that connect such systems with the physical world, leading to the *grounding problem* — see below). This makes the initial representation or encoding extremely important. If it is not properly chosen (by the designer of the system or by evolution), it places additional (and perhaps insurmountable) burdens on the learning mechanisms (e.g., if the initial representation failed to capture spatial or temporal relations at a level of detail that is necessary for dealing with the problem domain).

It is far from clear that every task-specific representation ever used by the system must be learned. Representations may be constructed as necessary to solve specific problems and then discarded. Alternatively, some basic (learned or evolved) representations may be adapted in real time for solving specific problems. This is an important aspect of the schema-based approach to modelling intelligence proposed by Arbib (1994).

As already pointed out, living brains appear to provide a rich panoply of representations — including analogical and iconic representations in the form of serial-parallel networks of topography-preserving projections (Kuffler, Nicholls, and Martin, 1984; Zeki and Shipp, 1988; Uhr, 1986; Honavar, 1989; Honavar and Uhr, 1989a; 1989b) in the visual, auditory and motor cortices. Such representations have been largely ignored in today's SAI as well as NANN models. They may very well be among the essential representations grounded in the environment that form the foundation of a much larger representational edifice that is needed for human-like general intelligence.

In short, SAI and NANN systems often differ in terms of the preferred form of knowledge representation used although any knowledge that can be represented in one can also be represented (albeit not as efficiently, robustly or elegantly) in the other. The challenge for engineers and cognitive modellers is to choose the right mix of SAI and NANN (and whatever other possibilities that exist) to meet the needs of the problem at hand.

4 A CLOSER LOOK AT SAI AND NANN

Given the shared philosophical and scientific roots that SAI and NANN have in common, why the great fuss about their integration? Answering this question entails taking a closer look at some prototypical SAI and NANN systems followed by a critical examination of what are generally considered their defining characteristics and much-touted advantages and disadvantages. This examination demonstrates that despite assertions by some to the contrary, the differences between them are less than what they might seem at first glance; and to the extent they differ, such differences are far from being in any reasonable sense of the term, fundamental; and that the purported weaknesses of each can potentially be overcome through a judicious integration of techniques and tools selected from the other (Honavar, 1990; Honavar and Uhr, 1990a; Uhr and Honavar, 1994; Honavar and Uhr, 1994; Uhr, 1990; Boden, 1994).

4.1 Problem Solving as State Space Search

State Space Search in SAI Systems

The prototypical SAI models are more or less direct descendents of the von Neumann stored program model of computation. The essential components of such a model are: a storage for programs (instructions for processing data), a processor for interpretation and execution of the program; and a (transient) working memory for receiving inputs, and holding intermediate results of processing. Learning programs have additional mechanisms for self-modification (i.e., the modification of the set of programs that they use).

The dominant paradigm for problem solving in SAI is *state space search* (Winston, 1992; Ginsberg, 1993). States represent snap-shots of the problem at various stages of its solution. Operators enable transforming one state into another. Typically, the states are represented using structures of symbols (e.g., lists). Operators transform one symbol structure (e.g., list, or a set of logical expressions) into another. The system's task is to find a path between two specified states in the state-space (e.g., the initial state and a specified goal, the puzzle and its solution, the axioms and a theorem to be proved, etc.).

In almost any non-trivial problem, a blind exhaustive search for a path will be impossibly slow, and there will be no known algorithm or a procedure for directly computing that path without resorting to search. As a result, much work in SAI has focused on the study of effective heuristic search procedures (Pearl, 1984). For example, SAI systems handle games like chess as follows: The initial board is established as the given, and a procedure is coded to compute whether a win-state has been reached. In addition, procedures are coded to execute legal moves and (usually) to compute heuristic assessments of the promise of each possible move, and to combine the separate heuristic assessments into an overall value that will be used to choose the next move. Finally, all these are put into a total structure that applies the appropriate heuristics, combines their results and evaluates alternative moves, and actually makes a move, then waits for and senses the opponent's moves, uses it to update the board (probably checking that it is indeed a legal move), and loops back to make its own next move. (For simplicity the look-ahead with minimax that most game-players use has been ignored, but that is essentially more of the same.)

Knowledge-Guided Search

Search in general can be guided by the knowledge that is at the disposal of the problem solver. If the system is highly specialized, the necessary knowledge is usually built into the search procedure (in the form of criteria for choosing among alternative paths, heuristic functions to be used, etc.). However, general purpose problem solvers also need to be able to retrieve problem-specific and perhaps even situation-specific knowledge to be used to guide the search during problem-solving. Indeed, such retrieval might itself entail search (albeit in a different space). Efficient, and flexible representations of such knowledge as well as mechanisms for their retrieval as needed during problem solving are, (although typically overlooked because most current AI systems are designed for very specialized, narrowly defined tasks), extremely important. This is an area where NANN or other implementations of content addressed memories and indexing schemes are especially worth exploring.

State Space Search in NANN Systems

The NANN system (a network of relatively simple processing elements, neurons, or nodes) is typically presented with an input pattern or initialized in a given starting state encoded in the form of a state vector each of whose elements corresponds to the state of a neuron in the network). It is designed or trained to output the correct response to each input pattern it receives (perhaps after undergoing a series of state updates determined by the rules governing its dynamic behavior). The input-output behavior of the network is a function of the network architecture, the functions computed by the individual nodes and parameters such as the weights.

For example, the solution of an optimization problem (traditionally solved using search) can be formulated as a problem of arriving at a state of a suitably designed network that corresponds to one of minimum energy (which is defined to correspond in some natural way to the optimality of the solution being sought). For an example of such an approach to theorem-proving, see (Pinkas, 1994). Ideally, the network dynamics are set up so as to accomplish this without additional explicit control. However, in practice, state updates in NANN systems are often controlled in a manner that is not much different from explicit control (as in sequential update of neurons in Hopfield networks (Hopfield, 1982) where only one neuron is allowed to change its state on any update cycle) to guarantee certain desired emergent behaviors). Indeed, a range of cognitive tasks do require selective processing of information that often necessitates the use of a variety of (albeit flexible and distributed) networks of controls that is presently lacking in most NANN models (Honavar and Uhr, 1990b). Many such control structures and processes are suggested by an examination of computers, brains, immune systems, and evolutionary processes.

In short, in both SAI and NANN systems, problem-solving involves state-space search; and although most current implementations tend to fall at one end of the spectrum or the other, it should be clear that there exists a space of designs that can use a mix of different state representations and processing methods. The choice of a particular design for a particular class of problems should primarily be governed by performance, cost, and reliability considerations for artificial intelligence applications and psychological and neurobiological plausibility for cognitive modelling.

4.2 Symbols, Symbol Structures, Symbolic Processes

Symbols

Knowledge representation as described earlier, generally implies the use of symbols at some level. The standard notion of a symbol is that it *stands for* something and when a symbol token appears within a symbolic expression carries the interpretation that the symbol stands for something within the context that is specified by its place in the expression. In general, a symbol serves as a surrogate for a body of knowledge that may need to be accessed and used in processing the symbol. And ultimately, this knowledge includes semantics or meaning of the symbol in the context in which it appears, including that provided by the direct or indirect grounding of the symbol structure in the external environment (Harnad, 1990).

Symbolic Processes

Symbolic processes are essentially transformations that operate on symbol structures to produce other symbol structures. Memory holds symbol structures that contain symbol tokens that can be modified by such processes. This memory can take several forms based on the time scales at which such modifications are allowed. Some symbol structures might have the property of determining choice and the order of application of transformations to be applied on other symbol structures. These are essentially the programs. Programs when executed — typically through the conventional process of compilation and interpretation and eventually — when they operate on symbols that are linked through grounding to particular effectors — produce behavior. Working memory holds symbol structures as they are being processed. Long-term memory, generally speaking, is the repository of programs and can be changed by addition, deletion, or modification of symbol structures that it holds.

Such a system can compute any Turing-computable function provided it has sufficiently large memory and its primitive set of transformations are adequate for the composition of arbitrarily symbol structures (programs) and the interpreter is capable of interpreting any possible symbol structure. This also means that any particular set of symbolic processes can be carried out by an NANN — provided it has potentially infinite memory, or finds a way to use its transducers and effectors to use the external physical environment to serve as its memory).

Knowledge in SAI systems is typically embedded in complex symbol structures such as lists (Norvig, 1992), logical databases (Genesereth and Nilsson, 1987), semantic networks (Quillian, 1968), frames (Minsky, 1975), schemas (Arbib, 1972; 1994), and manipulated by (often serial) procedures or inferences (e.g., list processing, application of production rules (Waterman, 1985), or execution of logic programs (Kowalski, 1977) carried out by a central processor that accesses and changes data in memory using addresses and indices.

It is often claimed that the NANN systems predominantly perform numeric processing in contrast to SAI systems which manipulate symbol structures.

Symbolic Processes in NANN systems

As already pointed out, NANN systems represent problem states using (typically binary) state vectors which are manipulated in a network of processors using (typically) numeric operations (e.g., weighted sums and thresholds). It is not hard to see that the numeric state vectors and transformations employed in such networks play an essential symbolic role although the rules of transformation may now be an emergent property of a large number of nodes acting in concert. In short, the formal equivalence of the various computational models guarantees that NANN can support arbitrary symbolic processes. It is not therefore surprising that several alternative mechanisms for variable binding and logical reasoning using NANN have been discovered in recent years. Some of these require explicit use of symbols (Shastri and Ajjanagadde, 1989); others resort to quasi-symbols that have some properties of symbols while not being actually symbols in their true sense (Pollack, 1990; Maclennan, 1994); still others use pattern vectors to encode symbols (Dolan and Smolensky, 1989; Smolensky, 1990; Sun, 1994a; Chen and Honavar, 1994). The latter approach to symbol processing is often said to use sub-symbolic encoding of a symbol as a pattern vectors each of whose components is insufficient in and of itself to identify the symbol in question (see the discussion on distributed representations below). In any case, most, if not all, of these proposals are implemented and simulated on general purpose digital computers, so none of the functions that they compute are outside the Turing framework.

4.3 Numeric Processing

Numeric processing, as the name suggests, involves computations with numbers. On the surface it appears that most NANN perform essentially numeric processing. After all, the formal neuron of McCulloch and Pitts computes weighted sum of its numeric inputs. And the *neurons* in most NANN models perform similar numerical computations. On the other hand, SAI systems predominantly compute functions over structures of symbols. But numbers are in fact symbols for quantities; and any computable function over numbers can be computed by symbolic processes. In fact, general purpose digital computers have been performing both symbolic as well as numeric processing ever since they were invented.

4.4 Analog Processing

It is often claimed that NANN perform *analog computation*. Analog computation generally implies the use of dynamical systems describable using continuous differential equations. They operate in continuous time, generally with physical entities such as voltages, currents, which serve as physical *analogs* of the quantities of interest. Thus soap bubbles, servomechanisms, and cell membranes can all be regarded as analog computers (Rajaraman, 1981).

Whether physically realizable systems are truly analog or whether analog system is simply a mathematical idealization of (extremely fine-grained) discrete system is a question that borders on the philosophical (e.g., are matter, time and space continuous or discrete?). However, some things are fairly clear. Most NANN are simulated on digital computers and compute in discrete steps and hence are clearly not analog. The few NANN models can be regarded as analog devices — e.g., the analog VLSI circuits designed and built by Carver Mead and colleagues (Mead, 1989) — are incapable of discrete symbolic computations (because of their inability to make all-or-none or discrete choices) (Maclennan, 1994) although they can approximate such computations. (For example, the stable states or attractors of such systems can be interpreted as identifiable discrete states).

Analog systems can be, and often are simulated on digital computers at the desired level of precision. However, this might involve a time-consuming iterative calculation to produce a result that could potentially be obtained almost instantaneously (and transduced using appropriate transducers) given the right analog device. Thus analog processing appears to be potentially quite useful in many applications (especially those that involve perceptual and motor behavior). It is possible that evolution has equipped living systems with just the right repertoire of analog devices that help them process information in this fashion. However, it is somewhat misleading to call such processing *computation* (in the sense defined by Turing) because it lacks the discrete combinatorial structure that is characteristic of all Turing-equivalent models of computation (Maclennan, 1994).

Whether analog processes play a fundamental role (beyond being part of grounding of representations) in intelligent systems remains very much an open question. It is also worth pointing out that digital computers can, and in fact do, make use of essentially analog devices such as transistors but they use only a few discrete states to support computation (in other words, the actual analog value is irrelevant so long as it lies withing a range that is distinguishable from some other range). And when embedded in physical environments, both SAI and NANN systems do encounter analog processes through sensors and effectors.

4.5 Compositionality and Systematicity of Representation

It has been argued by many e.g., Fodor and Pylyshyn, 1988) that *compositionality* and *systematicity* (structure sensitivity) of representation are essential for explaining mind. In their view, NANN are inadequate models of mind because NANN representations lack these essential properties. Compositionality is the property that demands that representations must possess an internal *syntactic structure* as a consequence of a particular method for composing complex symbol structures from simpler components. Systematicity requires the existence of processes that are sensitive to the syntactic structure. As argued by Sharkey and Jackson (1994), lack of compositionality is demonstrably true only for a limited class of NANN representations; and compositionality and systematicity in and of themselves are inadequate to account for cognition (primarily for lack of grounding or semantics). Van Gelder and Port (1994) have shown that several forms of compositionality can be found in NANN representations.

4.6 Grounding and Semantics

Many in the artificial intelligence and cognitive science research community agree on the need for *grounding* of symbolic representations through sensory (e.g., visual, auditory, tactile) transducers and motor effectors in the external environment on the one hand and the internal environment of needs, drives, and emotions of the organism (or robot) in order for such representations (which are otherwise devoid of any intrinsic meaning to the organism or robot) to be-come imbued with meaning or semantics (Harnad, 1990). Some have argued that NANN systems provide the necessary apparatus for grounding (Harnad, Hanson, and Lubin, 1994). It is important to realize that NANN as computational models do not provide physical grounding (as opposed to grounding in a simulated world of virtual reality) for representations any more than their SAI counterparts. It is only the physical systems with their physical substrate on which the representations reside that are capable of providing such grounding in physical reality when equipped with the necessary transducers and effectors. This is true irrespective of whether the system in question is a prototypical SAI system, or a prototypical NANN system, or a hybrid or integrated system.

4.7 Serial versus Parallel Processing

As pointed out earlier, most of today's SAI systems are serial programs that are executed on serial von Neumann computers. However, serial symbol manipulation is more an artifact of most current implementations of SAI systems than a necessary property of SAI. In parallel and distributed computers, memory is often locally available to the processors and even can be almost eliminated in data flow machines which model functional or applicative programs where data is transformed as it flows through processors or functions. Search in SAI systems can be, and often is, parallelized by mapping the search algorithm onto a suitable network of computers (Uhr, 1984; 1987; Hewitt, 1977; Hillis, 1985) with varying degrees of centralized or distributed control. Many search problems that arise in applications such as temporal reasoning, resource allocation, scheduling, vision, language understanding and logic programming can be formulated as constraint satisfaction problems which often lend themselves to solution using a mix of serial and parallel processing (Tsang, 1993).

Similarly, SAI systems using production rules can be made parallel by enabling many rules to be matched simultaneously in a data flow fashion (as in RETE pattern matching networks (Forgy, 1982)). Multiple matched rules may be allowed to fire and change the working memory in parallel as in parallel production systems (Uhr, 1979) and classifier systems (Holland, 1975) — so long as whenever two or more rules demand conflicting actions, arbitration mechanisms are provided to choose among the alternatives or resolve such conflicts at the sensory-motor interface. Such arbitration mechanisms can themselves be realized using serial, parallel (e.g., winner-take-all mechanism), or serial-parallel (e.g., pyramid-like hierarchies of decision mechanisms) networks of processes.

NANN systems with their potential for massive fine-grained parallelism of computation offer a natural and attractive framework for the development of highly parallel architectures and algorithms for problem solving and inference. Such systems are considered necessary by many researchers (Uhr, 1980; Feldman and Ballard, 1982) for tasks such as real-time perception. But SAI systems doing symbolic inference can be, and often are, parallelized, and certain inherently sequential tasks need to be executed serially. On any given class of problems, the choice of decomposition of the computations to be performed into a parallel-serial network of processes and their mapping onto a particular network of processors has to be made taking the cost and performance tradeoffs into consideration.

4.8 Knowledge Engineering Versus Knowledge Acquisition Through Learning

The emphasis in some SAI systems (especially the so-called knowledge-based expert systems (Waterman, 1985)) on knowledge engineering has led some to claim that SAI systems are, unlike their NANN counterparts, incapable of learning from experience. This is clearly absurd as even a cursory look at the current research in machine learning (Shavlik and Dietterich, 1990; Buchanan and Wilkins, 1993) and much early work in pattern recognition (Uhr, 1973; Fu, 1982; Miclet, 1986) shows. Research in SAI and closely related systems indeed have provided a wide range of techniques for deductive (analytical) and inductive (synthetic) learning. Learning by acquisition and modification of symbol structures almost certainly plays a major role in knowledge acquisition in humans who learn and communicate in a wide variety of natural languages (e.g., English) as well as artificial ones (e.g., formal logic, programming languages). While NANN systems with their micro-modular architecture offer a range of interesting possibilities for learning, for the most part, only the simplest parameter or weight modification algorithms have been explored to date (McClelland and Rumelhart, 1986 et almr86; Kung, 1993; Haykin, 1993). In fact, learning by weight modification alone appears to be inadequate in and of itself to model rapid and irreversible learning that is observed in many animals. Algorithms that modify networks through structural changes that involve the recruitment of neurons (Greenough and Bailey, 1988; Honavar, 1989; 1990; Honavar and Uhr, 1989a; 1989b; 1993; Kung, 1993; Grossberg, 1980) appear promising in this regard.

A detailed discussion of learning is beyond the scope of this chapter. Suffices it to point out that most forms of learning can be understood and implemented in terms of structures and processes for representing and reasoning with knowledge (broadly interpreted) and for memorizing the results of such inference in a form that lends itself to retrieval and use at a later time (Michalski, 1993). Thus any NANN or SAI or some hybrid architecture that is capable of performing inference and has memory for storing the results of inference for retrieval and use on demand can be equipped with the ability to learn. The interested reader is referred to (Honavar, 1994) for a detailed discussion of systems that learn using multiple strategies and representations. Additional examples of systems that combine NANN and SAI approaches to learning can be found in (Uhr, 1973; Holland, 1975; Honavar, 1992; 1994; Honavar and Uhr, 1993; Lacher and Nguyen, 1994; Carpenter and Grossberg, 1994; Shavlik, 1994; Goldfarb and Nigam, 1994; Booker, Riolo, and Holland, 1994). In short, SAI systems offer powerful mechanisms for manipulation of highly expressive structured symbolic representations while NANN offer the potential for robustness, and the ability to fine-tune their use as a function of experience (primarily due to the use of tunable numeric weights and statistics).

4.9 Associative as Opposed to Address-Based Storage and Recall

An often cited distinction between SAI and NANN systems is that the latter employ associative (i.e., content-addressable) as opposed to the address-and-index based storage and recall of patterns in memory typically used by the former. This is a misconception for several reasons: Address-and-index based memory storage and retrieval can be used to simulate content-addressable memory and vice versa and therefore unless one had access to the detailed internal design operation of such systems, their behavior can be indistinguishable from each other. Many SAI systems conventional computers use associative memories in some form or another (e.g., hierarchical cache memories). While associative recall may be better for certain tasks, address (or location-based) recall may be more appropriate for others. Indeed, many computational problems that arise in symbolic inference (pattern matching and unification in rule-based production systems or logic programming) can take advantage of associative memories for efficient processing (Chen and Honavar, 1994).

In prototypical NANN models, associative recall is based on some relatively simple measure of proximity or closeness (usually measured by Hamming distance in the case of binary patterns) to the stored patterns. While this may be appropriate in domains in which related items have patterns or codes that are close to each other, it would be absurd to blindly employ such a simple content-addressed memory model in domains where symbols are arbitrarily coded for storage (which would make hamming distance or a similar proximity measure useless in recalling the associations that are really of interest). Establishing (possibly context-sensitive) associations between otherwise arbitrary symbol structures based on their meanings and retrieving such associations efficiently requires complex networks of learned associations more reminiscent of associative knowledge networks, semantic networks (Quillian, 1968), frames (Minsky, 1975), conceptual structures (Sowa, 1984), schemas (Arbib, 1994), agents (Minsky, 1986) and object-oriented programs of SAI (Norvig, 1992) than today's simple NANN associative memory models. This is not to suggest that such structures cannot be implemented using suitable NANN building blocks — see (Arbib, 1994; Dyer, 1994; 1994b; Miikkulainen, 1994a; Bookman, 1994; Barnden, 1994a; 1994b) for some examples of such implementations. Indeed, such NANN implementations of complex symbol structures and symbolic processes can offer many potential advantages (e.g., robustness, parallelism) for SAI.

4.10 Distributed Storage, Processing, and Control

Distributed storage, processing, and control are often claimed to be some of the major advantages of NANN systems over their SAI counterparts. It is far from clear as to what is generally meant by the term *distributed* when used in this context (Oden, 1994).

Perhaps it is most natural to think of an item as distributed when it is coded (say as a pattern vector) whose components by themselves are neither sufficient to identify the item nor have any useful semantic content. Thus, the binary code for a letter of the alphabet is distributed. Any item thus distributed eventually has to be reconstructed from the pieces of its code. This form of distribution may be in space, time, or both. Thus the binary code for a letter of the alphabet may be transmitted serially (distributed in time) over a single link that can carry 1 bit of information at a time or in parallel (distributed in space) using a multi-wire bus. If a system employs such a mechanism for transmission or storage of data, it also needs decoding mechanisms for reconstructing the coded item at the time of retrieval. It is easy to see that this is not a defining property of NANN systems as it is found in even the serial von Neumann computers. In any event, both NANN as well as SAI systems can use such distributed coding of symbols. And, as pointed out by Hanson and Burr (1990), distributed coding in and of itself, offers no representational capabilities that are not realizable using a non-distributed coding.

In the context of NANN, the term *distributed* is often used to refer to storage of parts of an item in a unit where parts of other items also stored (for example, by superposition). Thus, each unit participates in storage of multiple items and each item is distributed over multiple units. (There is something disconcerting about this particular use of the term distributed in a technical sense: Clearly, one can invent a new name for whatever it is that a unit stores — e.g., a number whose binary representation has a '1' in its second place. Does the system cease to be distributed as a result?). It is not hard to imagine an analogous notion of distribution in time instead of space but it is also fraught with similar semantic difficulty.

The term *distributed* when used in the context of *parallel and distributed processing*, generally refers to the decomposition of a computational task into more or less independent pieces that are executed on different processors with little or no inter-processor communication (Uhr, 1984; 1987; Almasi and Gottlieb, 1989). Thus many processors may perform the same computation on pieces of the data (as in single-instruction-multiple data or SIMD computer architectures) or each processor may perform a different computation on the same data e.g., computation of various intrinsic properties of an image (as in multiple-instruction-single-data or MISD computer architectures), or a combination of both (as in multiple-instruction-multiple-data or MIMD computer architectures). Clearly, both NANN and SAI systems can take advantage of such parallel and distributed processing. The reader is referred to (Almasi and Gottlieb, 1989; Uhr, 1984; 1987) for examples.

4.11 Redundancy and Fault Tolerance

Often the term *distributed* is used more or less synonymously with *redundant* and hence fault-tolerant in the NANN literature. This is misleading because there are many ways to ensure redundancy of representation, processing and control. One of the simplest involves storing multiple copies of items and/or using multiple processors to replicate the same computation in parallel, and using a simple majority vote or more sophisticated statistical evidence combination processes to pick the result. Redundancy and distributivity are orthogonal properties of representations. And clearly, SAI as well as NANN systems can be made redundant and fault-tolerant using the same techniques.

4.12 Statistical, Fuzzy, or Evidential Inference

It is often claimed that NANN models provide noise-tolerant and robust inference because of the probabilistic, fuzzy, or evidential nature of the inference mechanisms used. This is largely due to combination and weighting of evidence from multiple sources through the use of numerical weights or probabilities. It is possible to establish the formal equivalence inference in certain classes of NANN models with probabilistic or fuzzy rules of reasoning. But fuzzy logic (Zadeh, 1975; Yager and Zadeh, 1993) operates (as its very name suggests), with *logical* (hence symbolic) representations. Probabilistic reasoning is an important and active area of research in SAI as well (See Pearl, 1988 for details). Heuristic evaluation functions that are widely used in many SAI systems provide additional examples of approximate, that is, not strictly truth-preserving inference in SAI systems. In many SAI systems, the requirements of soundness and completeness of inference procedures are often sacrificed in exchange for efficiency. In such cases, additional mechanisms are used to (after the fact) verify and if necessary, override the results of inference if they are found to conflict with other evidence.

Much research on human reasoning indicates that people occasionally draw inferences that are logically unsound (Johnson-Laird and Byrne, 1991). This suggests that although people may be capable of applying sound inference procedures, they probably take shortcuts when faced with limited computational or memory resources. Approximate reasoning under uncertainty is clearly an important tool that both SAI and NANN systems can potentially employ to effectively make rapid, usually reliable and useful, but occasionally fallible inferences in real time.

4.13 SAI and NANN As Models of Minds/Brains

Some of the SAI research draws its inspiration from (rather superficial) analogies with the mind and mental phenomena and in turn contributes hypotheses and models to the study of minds; Similarly, many NANN models draw their inspiration from (albeit superficial) analogies with the brain and neural phenomena and in turn contribute models that occasionally shed light on some aspects of brain function (Churchland and Sejnowski, 1992).

It is important to emphasize that neither today's SAI nor today's NANN have the monopoly on modelling minds and brains. Today's NANN models are at best, extremely simplified caricatures of biological neural networks (Shepherd, 1989; 1990; McKenna, 1994). Biological neurons and microcircuits of neurons provide computational primitives that are far more powerful than simple threshold or sigmoids that are used in most NANN models (Uhr, 1994). Brains display highly structured yet flexible organization into regions, layers, and modules that perform specialized functions (Kuffler, Nicholls and Martin, 1984; Zeki and Shipp, 1988). Such networks may be modelled by highly structured NANN models that organize the neurons into locally connected, topography preserving layers that are organized in loosely hierarchical fashion (Uhr, 1986; Honavar and Uhr, 1989a; 1989b; Honavar, 1992). Such structures appear to organize the networks of the brain in space (in ways that reflect the physics of the environment using networks of analog representations) and time (through the use of feedback loops with varying amounts of delay, networks of clocks and osciallators).

The brain appears to perform symbolic, numeric, as well as analog processing. The pulses transmitted by neurons are digital; the membrane voltages are analog (continuous); The molecular level phenomena that involve closing and opening of channels appears to be digital; The diffuse influence of neurotransmitters and hormones appear to be both analog and digital.

Changes in learning appear to involve both gradual changes of the sort modeled by the parameter changing or weight modification algorithms of todays NANN as well as major structural changes involving the recruitment of neurons and changes in network topology (Greenough and Bailey, 1988; Honavar, 1989; 1990; Honavar and Uhr, 1989a; 1989b; 1993). In fact, learning by weight modification alone appears to be inadequate in and of itself to model rapid and irreversible learning that is observed in many animals.

Also missing from most NANN models are elaborate control structures and processes of the sort found in brains including networks of oscillators that control timing. Perception, learning and control in brains appear to utilize events at multiple spatial and temporal scales (Grossberg, 1982). Additional processes not currently modelled by NANN systems include processes that include networks of markers that guide neural development, structures and processes that carry information that might be used to generate other network structures, and so on (Honavar and Uhr, 1990).

Clearly, living minds/brains are among the few examples of truly versatile intelligent systems that we have today. They are our existence proof that such systems are indeed possible. So even those whose primary interests are in constructing artificial intelligence systems can ill afford to ignore the insights offered by a study of biological intelligence (McKenna, 1994). (This does not of course mean that such an effort cannot exploit alternative technologies to accomplish the same functions, perhaps even better than their natural counterparts). But it is a misconception to assume that today's NANN model brains any more than today's SAI programs model minds. In short, the processes of the minds appear to be far less rigidly structured and far more flexible than today's SAI systems and the brains appear to have a lot more structure, organization, and control than today's homogeneous networks of simple processing elements which we call NANN. A rich space of designs that combine aspects of both within a well-designed architecture for intelligence remains to be explored.

5 INTEGRATION OF SAI AND NANN

It must be clear from the discussion in the previous sections that at least on the surface it looks like SAI and NANN are each appropriate, and possibly even necessary for certain problems, and grossly inappropriate, almost impossible, for others. But of course each can do anything that the other can. The issues are ones of performance, efficiency and elegance (and in cognitive modelling, perhaps plausibility in terms of the various known constraints between different levels — such as psychological, neurobiological, and neurochemical — at which satisfactory explanations are sought), and not theoretical capabilities as computational models.

This is a common problem in computing. One computer or programming language may be extremely well-suited for some problems but awful for others, while a second computer or language may be the opposite. This suggests several engineering possibilities (Uhr and Honavar, 1994), including:

1. Try to re-formulate and re-code the problem to better fit the computer or language.
2. Use one computer or language for some parts of the process and the other for others.
3. Build a new computer or language that contains constructs from each, and use these as appropriate.
4. Try to find as elegant as possible a set of primitives that underlie both computers or languages, and use these to build a new system.

The term *hybrid* is beginning to be used for systems that in some way try to combine SAI and NANN. If any of the above is called a hybrid probably all of the others should also. But usually hybrid refers to systems of type [2] or [3]. Types [3] and [4] would appear to be better than [2] (although harder to realize), since they would probably be more efficient and more elegant. Thus the capabilities of both SAI and NANN should be combined by tearing them apart to the essential components of their underlying processes and integrating these as closely as possible. Then the problem should be re-formulated and re-coded to fit this new system as well as possible. This restates a general principle most people are coming to agree on with respect to the design of multicomputer networks and parallel and distributed algorithms: the algorithm and the architecture should be designed to fit together as well as possible, giving algorithm-structured architectures and architecture-structured algorithms (Uhr, 1984; 1987; Almasi and Gottlieb, 1989).

6 SUMMARY

SAI and NANN each demonstrate at least one way of performing certain tasks naturally and thus pose the interesting problem for the other of doing something equivalent perhaps more elegantly, efficiently, or robustly than the other. It should be clear from the discussion above that the integration of SAI and NANN systems can be beneficially explored along several dimensions.

In the short term, *hybrid* architectures that use NANN and SAI modules to perform different but well-coordinated sets of functions in specific applications are definitely worth exploring. A partial list of examples of such integration include: neural network and expert knowledge based systems (Lin and Hendler, 1994; Shavlik, 1994; Gallant, 1993; Medsker, 1994); systems for language processing (Bookman, 1994; Dyer, 1994a; 1994b; Miikkulainen, 1994a; 1994b; Barnden, 1994b; Omlin and Giles, 1994; Smolensky, Legendre, and Miyata, 1994; Servan-Schreiber, Cleeremans, and McClelland, 1994); systems for visual pattern recognition and spatial reasoning (Honavar and Uhr, 1989a; 1989b; Honavar and Uhr, 1994; Honavar, 1994; Ballard and Brown, 1982; Uhr, 1987; Tanimoto and Klinger, 1980; Wechsler, 1990; Duda and Hart, 1973; Fu, 1982; Miclet, 1982; Carpenter and Grossberg, 1994; Kosslyn and Jacobs, 1994; Mjolsness, 1994); systems for symbolic inference (Sun, 1994a; 1994b; Barnden, 1994b; Smolensky, 1990; Shastri and Ajjanagadde, 1989; Chen and Honavar, 1994); systems for learning (Honavar and Uhr, 1989a; 1989b; 1993; Honavar, 1992; 1994; Shavlik, 1994; Goldfarb and Nigam, 1994; Fu, 1982; Fukunaga, 1990; Gallant, 1994; Uhr, 1973; Holland, 1975; Booker, Riolo, and Holland, 1994; Lacher and Nguyen, 1994; Carpenter and Grossberg, 1994; Dyer, 1994a). These efforts offer a number of important insights into the design and performance of such hybrid systems for cognitive modelling on the one hand and engineering intelligent systems for practical applications on the other (see below).

The integration of concepts, constructs, techniques and technologies drawn from SAI and NANN as well as other closely related paradigms (including statistical pattern recognition, syntactic pattern recognition, evolutionary computation) offers a rich and potentially very promising design space for exploration by artificial intelligence engineers and cognitive theorists. It is becoming increasingly obvious that this space exhibits almost infinite variety that is characteristic of complex systems. In the long-term, a coherent theoretical framework for analysis and synthesis of such systems has to be developed. In order to do this, we need to start developing and refining our categorization of such mutually inter-related systems. One way to approach this task is to seek categorizations that capture essential underlying principles of the *architecture*, alternative *implementations* of the architecture, and finally alternative or *physical realization* of the candidate implementations of such systems relative to our goals of understanding and engineering intelligence.

Because of the engineering and technological emphasis of artificial intelligence, most research in the area has focused on the development of algorithms for specific tasks that appear to require intelligence if performed by humans (e.g., diagnosis, planning, character recognition). While such efforts provide useful technological tools in the short term, they appear to have fallen short of providing much insight into alternative implementations and physical realizations of architectures for general intelligence.

Most artificial intelligence and cognitive science theories of intelligence are primarily about the content of knowledge or types of knowledge for some task of interest, with minimal commitment on the choice of architecture (or equivalently, the programming language that defines the virtual architecture of the computer). Perhaps this is because it is tacitly assumed that any such architecture is one that is capable of supporting universal computation and that nothing else about it is of much interest. Perhaps this is where the dichotomy between SAI and NANN can help focus our attention on architectural issues. After all, NANN models do (in most cases) represent architectural commitment(s) that are different from those implicitly assumed by SAI models (e.g., lambda calculus or production systems). However, it must be emphasized that one architectural commitment is not necessarily better than another independent of the task for which the architecture is used. Also worth noting is the fact that the same system may be lend itself to multiple architectural descriptions. Each such description can potentially add to our understanding of different aspects of the system in important ways. Furthermore, each architectural description lends itself to multiple implementations; For example, the same architecture can be implemented using a network of simple processors or simulated by a program on a conventional serial computer. And each implementation lends itself to multiple physical realizations.

Living minds/brains offer an existence proof of at least one architecture for general intelligence. SAI and NANN paradigms together offer a wide range of architectural choices. Each architectural choice brings with it some obvious (and some not so obvious) advantages as well as disadvantages in the solution of specific problems using specific algorithms, given certain performance demands and design constraints imposed by the available choices of physical realizations of the architecture. Together, the cross-product of the space of architectures, algorithms, and physical realizations constitutes a large and interesting space of possible designs for intelligent systems. Examples of systems resulting from a judicious integration of concepts, constructs, techniques and technologies drawn from both traditional artificial intelligence systems and artificial neural networks clearly demonstrate the potential benefits of exploring this space. And, perhaps more importantly, the rather severe practical limitations of today's SAI and NANN systems strongly argues for the need for a systematic exploration of such design space.

This suggests that it might be fruitful to approach the choice of architectures, implememtations, and their physical realizations using the entire armamentarium of tools drawn from the theory and practice of computer science — including the design of programming languages (and hence virtual architectures), computers, algorithms, and programs. Our primary task is to identify subsets of Turing-computable functions necessary for general intelligence, an appropriate mix of architectures for supporting specific subsets of these functions, as well as appropriate realizations of such architectures in physical devices. The hybrid or integrated SAI-NANN designs explored to date — including those examined in several recent books on this subject (Honavar and Uhr, 1994a; Sun and Bookman, 1994; Goonatilake and Khebbal, 1994; Levine and Aparicioiv, 1994) are only suggestive of a much larger space of interesting possibilities. It is almost certainly premature to pick one architecture over another as the architecture of choice for general intelligence (of the sort attributed to humans), or even eliminate certain architectures from consideration as candidates. Such choices can be made only after a careful evaluation of possible designs.

Acknowledgements

This work was partially supported by the Iowa State University College of Liberal Arts and Sciences and the National Science Foundation grant IRI-9409580. The author is indebted to Professors Leonard Uhr and Gregg Oden for extensive discussions on the topic of this chapter. He would like to thank Dr. Ron Sun and Dr. Larry Bookman for their invitation to him to contribute to this book as well as their comments on an earlier draft of this chapter.

REFERENCES

[1] Almasi, G.S. and Gottlieb, A. (1989). *Highly Parallel Computing.* New York: Benjamin-Cummings.

[2] Arbib, M.A. (1972). *The Metaphorical Brain.* New York: Wiley-Interscience.

[3] Arbib, M.A. (1994). Schema Theory: Cooperative Computation for Brain Theory and Distributed AI. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[4] Ballard, D. and Brown, C. (1982) *Computer Vision.* Englewood Cliffs, NJ: Prentice Hall.

[5] Barnden, J.A. (1994a). How Might Connectionist Networks Represent Propositional Attitudes? In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[6] Barnden, J.A. (1994b). Complex Symbol Processing in a Transiently Localist Connectionist Architecture. In: *Computational Architectures Integrating Symbolic and Neural Processes.* Sun, R. and Bookman, L.A. (Ed.) Boston: Kluwer.

[7] Bickhard, M.H. (1993). Troubles With Computationalism. (draft)

[8] Boden, M. (1994). Horses of a Different Color? In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[9] Booker, L.E., Riolo, R.L., and Holland, J.H. (1994). Learning and Representation in Classifier Systems. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[10] Bookman, L.A. (1994). A Framework for Integrating Relational and Associational Knowledge for Comprehension. In: *Computational Architectures Integrating Symbolic and Neural Processes.* Sun, R. and Bookman, L.A. (Ed.) Boston: Kluwer.

[11] Buchanan, B.G. and Wilkins, D.C. (1993). *Readings in Knowledge Acquisition and Learning.* San Mateo, CA: Morgan Kaufmann.

[12] Carpenter, G. and Grossberg, S. (1994). Integrating Symbolic and Neural Processes in a Self-Organizing Architecture for Pattern Recognition and Prediction. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[13] Chandrasekaran, B. and Josephson, S.G. (1994). Architecture of Intelligence: The Problems and Current Approaches to Solutions. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[14] Chen, C. and Honavar, V. (1994). Neural Networks for Inference. In preparation.

[15] Churchland, P.S. (1986). *Neurophilosophy.* Cambridge, MA: MIT Press.

[16] Churchland, P.S. and Sejnowski, T.J. (1992). *The Computational Brain.* Boston, MA: MIT Press.

[17] Cohen, D. (1986). *Introduction to Computer Theory.* New Tork: Wiley.

[18] Cooper, E.D. (1990). An object-oriented interpreter for symbol train processing. (Preprint).

[19] Dolan, C.P. and Smolensky, P. (1989). Tensor product production system: A modular architecture and representation. *Connection Science*, 1:53-58.

[20] Duda, R.O. and Hart, P.E. (1973). *Pattern Classification and Scene Analysis.* New York: Wiley.

[21] Dyer, M. (1994). Grounding Language in Perception. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[22] Feigenbaum, E.A. (1963). In: *Computers and Thought.* Feigenbaum, E.A. and Feldman, J. (Ed.) New York: McGraw-Hill.

[23] Feldman, J.A. and Ballard, D.H. (1982). Connectionist models and their properties. *Cognitive Science*, 6:205-264.

[24] Fodor, J. (1976). *The Language of Thought.* Boston, MA: Harvard University Press.

[25] Fodor, J. and Pylyshyn, Z.W. (1988). Connectionism and cognitive architecture: A critical analysis. In: *Connections and Symbols.* Pinker, S. and Mehler, J. (Ed.) Cambridge, MA: MIT Press.

[26] Forgy, C.L. (1982). RETE: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem. *Artificial Intelligence*, 19:17-37.

[27] Fu, K.S. (1982). *Syntactic Pattern Recognition and Applications.* Englewood Cliffs, NJ: Prentice Hall.

[28] Fukunaga, K. (1990). *Introduction to Statistical Pattern Recognition.* New York: Academic Press.

[29] Gallant, S. (1993). *Neural Networks and Expert Systems.* Cambridge, MA: MIT Press.

[30] Genesereth, M.R., and Nilsson, N.J. (1987). *Logical Foundations of Artificial Intelligence.* Palo Alto, CA: Morgan Kaufmann.

[31] Ginsberg, M. (1993). *Essentials of Artificial Intelligence.* San Mateo, CA: Morgan Kaufmann.

[32] Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* Reading, MA: Addison-Wesley.

[33] Goldfarb, L. and Nigam, S. (1994). The Unified Learning Paradigm: A Foundation for AI. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[34] Goonatilake, S. and Khebbal, S. (1994). (Ed.) *Hybrid Intelligent Systems.* London: Wiley.

[35] Greenough, W.T. and Bailey, C.H. (1988). The Anatomy of Memory: Convergence of Results Across a Diversity of Tests. *Trends in Neuroscience* 11, 142-147.

[36] Grossberg, S. (1982). *Studies of Mind and Brain.* Boston, MA: Reidel.

[37] Hanson, S.J. and Burr, D. (1990). What Connectionist Models Learn: Learning and Representation in Connectionist Networks. *Behavior and Brain Sciences*, 13:1-54.

[38] Harnad, S. (1990). The Symbol Grounding Problem. *Physica D*, 42:335-346.

[39] Harnad, S., Hanson, S.J., and Lubin, J. (1994). Learned Categorical Perception in Neural Nets: Implications for Symbol Grounding. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[40] Haykin, S. (1994). *Neural Networks.* New York: Macmillan.

[41] Hewitt, C. (1977). Viewing Control Structures as Patterns of Passing Messages. *Artificial Intelligence*, 8:232-364.

[42] Hillis, D. (1985). *The Connection Machine.* Cambridge, MA: MIT Press.

[43] Holland, J.H. (1975). *Adaptation in Natural and Artificial Systems.* Ann Arbor, MI: University of Michigan Press.

[44] Hopfield, J.J. (1982). Neural Networks and Physical Systems With Emergent Collective Computational Abilities. *Proceedings of the National Academy of Sciences*, 79:2554-2558.

[45] Honavar, V. (1989). Perceptual Development and Learning: From Behavioral, Neurophysiological and Morphological Evidence to Computational Models. Tech. Rep. 818. Computer Sciences Dept., University of Wisconsin, Madison, Wisconsin.

[46] Honavar, V. (1990). *Generative Learning Structures for Generalized Connectionist Networks.* Ph.D. Dissertation. University of Wisconsin, Madison, Wisconsin.

[47] Honavar, V. (1992). Inductive Learning Using Generalized Distance Measures. In: *Proceedings of the SPIE Conference on Adaptive and Learning Systems.* Orlando, Florida.

[48] Honavar, V. (1994). Toward Learning Systems That Use Multiple Strategies and Representations. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[49] Honavar, V. and Uhr, L. (1989a). Brain-Structured Connectionist Networks that Perceive and Learn. *Connection Science*, 1:139-159.

[50] Honavar, V. and Uhr, L. (1989b). Generation, Local Receptive Fields, and Global Convergence Improve Perceptual Learning in Connectionist Networks. In: *Proceedings of the 1989 International Joint Conference on Artificial Intelligence*, San Mateo, CA: Morgan Kaufmann.

[51] Honavar, V. and Uhr, L. (1990a). Symbol Processing Systems, Connectionist Networks, and Generalized Connectionist Networks. Tech. Rep. 90-23. Department of Computer Science, Iowa State University, Ames, Iowa.

[52] Honavar, V. and Uhr, L. (1990b). Coordination and Control Structures and Processes: Possibilities for Connectionist Networks. *Journal of Experimental and Theoretical Artificial Intelligence*, 2:277-302.

[53] Honavar, V. and Uhr, L. (1993) Generative Learning Structures and Processes for Generalized Connectionist Networks, *Information Sciences*, 70:75-108.

[54] Honavar, V. and Uhr, L. (1994a). (Ed.) *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* San Diego, CA.: Academic Press.

[55] Honavar, V. and Uhr, L. (1994b). Toward Integrated Architectures for Artificial Intelligence and Cognitive Modelling. In: *Intelligent Hybrid Systems.* Goonatilake, S. and Khebbal, S. (Ed). London: Wiley.

[56] Johnson-Laird, P. and Byrne, J. (1991). *Deduction.* New York: Lawrence Erlbaum.

[57] Klir, G.J. (1969). *An Approach to General Systems Theory.* New York: Van Nostrand Reinhold.

[58] Klir, G.J. (1985). *Architecture of Systems Problem Solving.* New York: Plenum.

[59] Kosslyn, S. and Jacobs, R.A. (1994). Encoding Shape and Spatial Relations: A Simple Mechanism for Coordinating Multiple Representations. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[60] Kowalski, R.A. (1977). *Predicate Logic as a Programming Language.* Amsterdam: North-Holland.

[61] Koza, J. (1992). *Genetic Programming.* Boston, MA: MIT Press.

[62] Kuffler, S.W., Nicholls, J.G., and Martin, A.R. (1984). *From Neuron to Brain.* Sunderland, MA: Sinaur.

[63] Kung, S. Y. (1993). *Digital Neural Networks.* New York: Prentice Hall.

[64] Lacher, R.C. and Nguyen, K.D. (1994). Hierarchical Architectures for Reasoning. In: *Computational Architectures Integrating Symbolic and Neural Processes.* Sun, R. and Bookman, L.A. (Ed.) Boston: Kluwer.

[65] Levine, D.S. and Aparicioiv, M. (1994). (Ed.) *Neural Networks for Knowledge Representation.* New York: Lawrence Erlbaum.

[66] Lin, C. and Hendler, J. (1994). A Study of a Hybrid Connectionist-Symbolic System for The Analysis of Ballistic Signals. In: *Computational Architectures Integrating Symbolic and Neural Processes.* Sun, R. and Bookman, L.A. (Ed.) Boston: Kluwer.

[67] Maclennan, B.J. (1994). Image and Symbol — Continuous Computation and the Emergence of the Discrete. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[68] McCulloch, W.S. and Pitts, W. (1943). A Logical Calculus of the Ideas Immanent in Neural Activity. *Bulletin of Mathematical Biophysics*, 5:115-137.

[69] McClelland J., Rumelhart, D. and the PDP Research Group (1986). (Ed.) *Parallel Distributed Processing.* Boston, MA: MIT Press.

[70] McKenna, T. (1994). The Role of Inter-Disciplinary Research Involving Neuroscience in the Design of Intelligent Systems. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[71] Mead, C. (1989). *Analog VLSI and Neural Systems.* Reading, MA: Addison-Wesley.

[72] Medsker, L. (1994). *Hybrid Neural Network and Expert Systems.* Boston: Kluwer.

[73] Miikkulainen, R. (1994a). Integrated Connectionist Models: Building AI Systems on Subsymbolic Foundations. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[74] Miikkulainen, R. (1994b). Subsymbolic Parsing of Embedded Structures. In: *Computational Architectures Integrating Symbolic and Neural Processes.* Sun, R. and Bookman, L.A. (Ed.) Boston: Kluwer.

[75] Michalewicz, Z. (1992). *Genetic Algorithms + Data Structures = Evolution Programs.* New York: Springer-Verlag.

[76] Michalski, R.S. (1993). Toward a Unified Theory of Learning: Multi-Strategy Task-Adaptive Learning. In: *Readings in Knowledge Acquisition and Learning.* Buchanan, B.G., and Wilkins, D.C. (Ed.) San Mateo, CA: Morgan Kaufmann.

[77] Miclet, L. (1986). *Structural Methods in Pattern Recognition.* New York: Springer-Verlag.

[78] Minsky, M. (1963). Steps Toward Artificial Intelligence. In: *Computers and Thought.* Feigenbaum, E.A. and Feldman, J. (Ed.) New York: McGraw-Hill.

[79] Minsky, M. (1975). A Framework for Representing Knowledge. In: *The Psychology of Computer Vision.* Winston, P. H. (Ed). New York: McGraw-Hill.

[80] Minsky, M. (1986). *Society of Mind.* New York: Basic Books.

[81] Mjolsness, E. (1994). Connectionist Grammars for High-Level Vision. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[82] Narendra, K. S. and Annaswamy, A. M. (1989). *Stable Adaptive Systems.* Englewood Cliffs, NJ: Prentice-Hall.

[83] Newell, A. (1980). Symbol Systems. *Cognitive Science*, 4:135-183.

[84] Newell, A. (1990). *Unified Theories of Cognition.* Cambridge, MA: Harvard University Press.

[85] Newell, A., Shaw, J.C., and Simon, H. (1963). In: *Computers and Thought.* Feigenbaum, E.A., and Feldman, J. (Ed.) New York: McGraw-Hill.

[86] Norman, D. A. (1986). Reflections on Cognition and Parallel Distributed Processing. In: *Parallel Distributed Processing.* McClelland, J., Rumelhart. D. and the PDP Research Group (Ed.). Cambridge, MA: MIT Press.

[87] Norvig, P. (1992). *Paradigms in Artificial Intelligence Programming.* Palo Alto, CA: Morgan Kaufmann.

[88] Oden, G.C. (1994). Why the Difference Between Connectionism and Anything Else is More Than You Might Think But Less Than You Might Hope. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[89] Omlin, W.C. and Giles, C.L. Extraction and Insertion of Symbolic Information in Recurrent Neural Networks. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[90] Pearl, J. (1984). *Heuristics.* New York: Addison-Wesley.

[91] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems.* Palo Alto, CA: Morgan Kaufmann.

[92] Pinkas, G. (1994). A Fault-Tolerant Connectionist Architecture for the Construction of Logic Proofs. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[93] Pollack, J. (1990). Recursive Distributed Representations. *Artificial Intelligence*, 46:77-105.

[94] Quillian, M.R. (1968). Semantic Memory. In: *Semantic Information Processing.* Minsky, M. (Ed.) Cambridge, MA: MIT Press.

[95] Rajaraman, V. (1981). *Analog Computation and Simulation.* Englewood Cliffs: Prentice Hall.

[96] Rashevsky, N. (1960). *Mathematical Biophysics.* New York: Dover.

[97] Rosenblatt, F. (1962). *Principles of Neurodynamics.* Washington, DC: Spartan.

[98] Schneider, W. (1987). Connectionism: Is it a paradigm shift for psychology? *Behavior Research Methods, Instruments, and Computers*, 19:73-83.

[99] Selfridge, O.G. and Neisser, U. (1963). Pattern Recognition by Machine. In: *Computers and Thought.* Feigenbaum, E.A. and Feldman, J. (Ed.) New York: McGraw-Hill.

[100] Servan-Schreiber, D., Cleeremans, A., and McClelland, J. (1994). Graded State Machines: Representation of Temporal Contingencies in Recurrent Neural Networks. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[101] Sharkey, N. and Jackson, S.J. (1994). Three Horns of the Representational Dilemma. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[102] Shastri, L. and Ajjanagadde, V. (1989). Connectionist System for Rule Based Reasoning with Multi-Place Predicates and Variables. Tech. Rep. MS-CIS-8906. Computer and Information Science Dept. University of Pennsylvania, Philadelphia, PA.

[103] Shavlik, J.W. and Dietterich, T.G. (1990). (Ed). *Readings in Machine Learning.* San Mateo, California: Morgan Kaufmann.

[104] Shavlik, J.W. (1994). A Framework for Combining Symbolic and Neural Learning. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[105] Shepherd, G.M. (1989). The significance of real neuron architectures for neural network simulations. In: *Computational Neuroscience.* Schwartz, E. (Ed.) Cambridge, MA: MIT Press.

[106] Shepherd, G.M. (Ed.) (1990). *Synaptic Organization of the Brain.* New York, NY: Oxford University Press.

[107] Smolensky, P. (1990). Tensor Product Variable Binding and the Representation of Symbolic Structures in Connectionist Systems. *Artificial Intelligence*, 46:159-216.

[108] Smolensky, P., Legendre, G., and Miyata, Y. (1994). Integrating Connectionist and Symbolic Computation for the Theory of Language. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[109] Sowa, J.F. (1984). Conceptual Structures: Information Processing in Mind and Machine. Reading, Massachusetts: Addison-Wesley.

[110] Sun, R. (1994a). Logic and Variables in Connectionist Models: A Brief Overview. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[111] Sun, R. (1994b). A Two-level Architecture for Commonsense Reasoning. In: *Computational Architectures Integrating Symbolic and Neural Processes.* Sun, R. and Bookman, L.A. (Ed.) Boston: Kluwer.

[112] Sun, R. and Bookman, L.A. (1994). (Ed.) *Computational Architectures Integrating Symbolic and Neural Processes.* Boston: Kluwer.

[113] Turing, A.M. (1950). Computing Machinery and Intelligence. *Mind*, 59:433-460.

[114] Tanimoto, S.L. and Klinger, A. (1980). *Structured Computer Vision: Machine Perception Through Hierarchical Computation Structures.* New York: Academic Press.

[115] Tsang, E. (1993). *Foundations of Constraint Satisfaction.* New York: Academic Press.

[116] Uhr, L. and Vossler, C. (1963). A Pattern Recognition Program that Generates, Evaluates, and Adjusts its Own Operators. In: *Computers and Thought.* Feigenbaum, E. and Feldman, J. (Ed). New York: McGraw-Hill.

[117] Uhr, L. (1973). *Pattern Recognition, Learning, and Thought.* New York: Prentice-Hall.

[118] Uhr, L. (1979). Parallel-serial production systems with many working memories. In: *Proceedings of the Fifth International Joint Conference on Artificial Intelligence.*

[119] Uhr, L. (1980). In: *Structured Computer Vision.* Tanimoto, S., and Klinger, A. (Ed.) New York: Academic Press.

[120] Uhr, L. (1984). *Algorithm-Structured Computer Arrays and Networks.* New York: Academic Press.

[121] Uhr, L. (1986). Toward a Computational Information Processing Model of Object Perception. Tech. Rep. 651. Computer Sciences Dept., University of Wisconsin, Madison, Wisconsin.

[122] Uhr, L. (1987). *Multi-computer Architectures for Artificial Intelligence.* New York: Wiley.

[123] Uhr, L. (1990). Increasing the Power of Connectionist Networks by Improving Structures, Processes, Learning. *Connection Science*, 2:179-193.

[124] Uhr, L. (1994). Digital and Analog Sub-Net Structures for Connectionist Networks. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[125] Uhr, L. and Honavar, V. (1994). Artificial Intelligence and Neural Networks: Steps Toward Principled Integration. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[126] van Gelder, T. and Port, R. (1994). Beyond Symbolic: Prolegomena to a Kama-Sutra of Compositionality. In: *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration.* Honavar, V. and Uhr, L. (Ed.) San Diego, CA: Academic Press.

[127] Waterman, D.A. (1985). *A Guide to Expert Systems.* Reading, MA: Addison-Wesley.

[128] Wechsler, H. (1990). *Computational Vision.* New York: Academic Press.

[129] Winston, P.H. (1992). *Artificial Intelligence.* Boston, MA: Addison-Wesley.

[130] Yager, R.R. and Zadeh, L.A. (1994). (Ed.) *Fuzzy Sets, Neural Networks, and Soft Computing.* New York: Van Nostrand Reinhold.

[131] Zadeh, L.A. (1975). Fuzzy Logic and Approximate Reasoning. *Synthese*, 30:407-28.

[132] Zeidenberg, M. (1989). *Neural Networks in Artificial Intelligence.* New York: Wiley.

[133] Zeki, S. and Shipp, S. (1988). The Functional Logic of Cortical Connections. *Nature*, 335:311-317.

IOWA STATE UNIVERSITY

OF SCIENCE AND TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

Tech Report: TR94-14

Submission Date: August 18, 1994