A representation of representation applied to a discussion of variable binding *

Richard Rohwer
Dept. of Computer Science and Applied Mathematics
Aston University, Birmingham B4 7ET, UK

26 August 1992

Abstract

States or state sequences in neural network models are made to represent concepts from applications. This paper motivates, introduces and discusses a formalism for denoting such representations; a representation for representations. The formalism is illustrated by using it to discuss the representation of variable binding and inference abstractly, and then to present four specific representations. One of these is an apparently novel hybrid of phasic and tensor-product representations which retains the desirable properties of each.

1 Introduction

Can neural network models easily represent nontrivial aspects of cognition? This fundamental issue of connectionism is important from the psychological, philosophical and software engineering points of view. Unfortunately the question is rather ambiguous. It is easily demonstrated that a simple neural network model can emulate a finite state machine (It suffices to prescribe neural circuits for FLIP-FLOPs and NAND gates.), or even an (infinite-tape) Turing machine [4], so there is a sense in which a neural network model has as much representational power as any other computational model which is likely to be on offer. Thus the issue involves largely subjective judgements of which representations are 'easier' or more 'natural' than which.

A further ambiguity arises from imprecise and incomplete knowledge of cognition itself. Without a pre-existing accepted representation of cognition, how can any representation be judged for validity, let alone naturalness? Fortunately, it is possible to represent some nontrivial aspects of cognition formally, though the formalism may not reflect psychological reality with great accuracy. The predicate calculus, for example, serves as a formal representation of logical reasoning. Although normal human or animal reasoning may depart

*To appear in Proc. Neurodynamics and Psychology Workshop, 22-24 April 1992, Bangor, Wales, M. Oaksford and G. Brown, Eds.

substantially from this formalism, it captures something of human cognition because humans *can* reason this way, whether or not they normally do. Be that as it may, improved machine representations of this type of reasoning are likely to have practical utility.

This paper mainly concerns the validity of representations, avoiding the subjective questions of naturalness. That is, it is concerned with whether a representation illustrates the distinctions required. In addition to its use for validating putative representations it also aids understanding by accentuating the commonality between different representations of the same thing. Representations are normally invented in an *ad hoc* manner to suit the convenience of the problem at hand, or because a similarity between psychological data and a behaviour of a network model has been noticed. The formalism suggested here also opens the possibility of defining algorithms to automatically search for representations, which in turn may provide further material for the 'naturalness' debate as well as practical techniques.

First, a general formalism for representation is introduced and discussed. In essence it is a way of specifying a correspondence between patterns of activity in neural network models and patterns of cognition. Each of these must be formally represented to begin with, which is fine for the neural patterns but unfortunate for the cognitive ones. Thus, a design criterion for this formalism is to impose as little structure as possible on the cognitive patterns. The formalism is then illustrated by checking the validity of a few representations of variable binding and inference. Finally some remarks are made on a 'grey area' between validity checking and naturalness evaluation.

2 Representation

2.1 The setting

A neural network model is a dynamical system; it has a state which varies in time according to a dynamical law. An adequate example for this paper is a set of N nodes each of which assumes a value of 0 or 1 at each step of discrete time. Let the output of the $\underline{i^{th}}$ node at time t be called $y_i(t)$. The *state* of the network at time t is the set of node values $\mathbf{y}(t) = \{y_i(t)\}_{i=1}^{i=N}$. Let us refer to a finite sequence of states $\{\mathbf{y}(t)\}_{t-T+1}^{t}$ as a *state excursion* or just an *excursion*. The network state evolves under the action of a time-development operator F^τ: $F^\tau \mathbf{y}(t) = \mathbf{y}(t+\tau)$. Let Υ_t designate the excursion ending at time t (if the length T of the excursion is either understood from context or not of interest). Then with the evolution of excursions defined in the natural way, namely $F^\tau \Upsilon_t = \{F^\tau \mathbf{y}(t)\}_{t-T+1}^{t}$, one has $F^\tau \Upsilon_t = \Upsilon_{t+\tau}$.

2.2 The definition

Training a network can be viewed as adjusting the dynamical law in such a way that at least part of the resulting trajectory of the system state represents a computation of interest. For example, a network designer might adopt a convention by which assertions of truth values for certain propositions are associated with particular numerical values on certain nodes, and then engineer the dynamical law so that representations of correct truth values tend to appear. Or more boldly, a psychologist might hypothesise that particular firing

patterns correspond to particular mental experiences. For a word, let us refer to the entities represented as *experiences*, even when they are defined *ad hoc* for engineering purposes and have no plausible connection conscious mental experience. A *representation*, then, is some sort of correspondence between *excursions* and *experiences*.

It is best to avoid imposing any more structure than necessary on the wooly concept of experience. The minimal necessity is to be able to mention an experience of interest by name, and at least in some cases, to determine whether a given excursion represents that experience. Therefore let us require that any experience to be represented have an associated *test map* from the set of all possible excursions to a set of two elements; one to signify that the experience is represented and one to signify nothing. Specifically, any experience A has a test map, which might as well also be called A (the meaning being clear from context). A can be applied to any excursion Υ with the significance:

$$\begin{aligned} A(\Upsilon) &= 1 \quad \text{means '}\Upsilon\text{ represents }A\text{'} \\ A(\Upsilon) &= 0 \quad \text{means nothing.} \end{aligned} \tag{1}$$

That is, for each experience of interest, a test which can be applied to any neural state excursion will respond to at least some of those excursions which represent the experience. If the null response can be taken to signify that the excursion does not represent the experience, let us say that the test is *exhaustive*.

2.3 Discussion

A possible alternative to using test maps is to introduce a mapping directly from excursions to the represented experiences or vice versa. The former direction is undesirable because it is better to put experiences in the domain of a map than the range. If they were 'produced' in a range, then the mapping, together with the excursions, might impose some structure on them. In a domain they need only be mentioned, and then only if they are of interest. In either direction, a direct mapping is likely to be one-to-many, so it would have to be defined in a complicated way, perhaps in terms of many single-valued maps. The test maps can be employed for this purpose if desired: The set of states which represent experience A includes $A^{-1}(1)$, and the set of experiences represented by Υ includes $\{A|A(\Upsilon) = 1\}$. If the test is exhaustive, these sets define the direct maps in each direction; otherwise they provide a partial definition which may still be better than none.

By allowing test maps to be non-exhaustive, one avoids having to specify every excursion which represents an experience. The price for this is that separate test maps must be introduced if it is desired to test for non-representation (absence) of an experience. If A is exhaustive, then an adequate test $\overline{A}$ for non-representation of A by Υ is $1 - A(\Upsilon)$. Only exhaustive tests will be used in the examples considered here, but the formalism allows one to avoid declaring the representational significance of every excursion. In a sensible software design, corresponding presence and absence tests would never both respond positively to the same excursion, but this formalism does not automatically enforce this because there is no law against mental states representing logical contradictions.

Thus far the representation formalism does not consider 'degrees' or 'qualities' of representation. One might like to say that some excursions represent a given experience more or less strongly than others. This flexibility might be added by extending the range of the test

maps to real numbers, or to some class of statements about the quality of a representation. But in light of the earlier comments about putting vague concepts in the range of a mapping, it might be better to put them in the domain, and keep the test map ranges Boolean.

3 Representation of variable binding and inference

In this section, variable binding and inference are defined, and test maps are used to define what must be done to represent them. Examples follow. All the examples conform to a scheme which is first defined in terms of test maps alone.

3.1 Variable binding

Variable binding is the idea that a variable x is instantiated by a constant c, in which case one says that x is bound to c, and may write $x = c$. Normally a variable is bound to at most one constant, though one constant might be bound to many variables. Thus there are $(C+1)^V$ possible bindings of V variables to C constants, because each variable may be bound to exactly one of the C constants or be unbound. The number of excursions of length T in an N-node network is 2^{NT} (for binary-valued nodes), so a minimum requirement for representing any possible binding without confusion is $(C+1)^V \leq 2^{NT}$.

A representation of variable binding requires a binding test map B_{xc} for each variable x and constant c; $B_{xc}(\Upsilon) = 1$ signifies that Υ represents that x is bound to c. Attention here will be restricted to exhaustive representations, so the non-representation test is $\overline{B}_{xc}(\Upsilon) = 1 - B_{xc}(\Upsilon)$.

Note that if $B_{xc}(\Upsilon) = 1$ and $B_{x'c'}(\Upsilon) = 1$, it is not automatically justified to conclude that Υ also represents the conjunction $(x = c) \wedge (x' = c')$. A separate test function must be defined to test for representation of conjunctions. After all, conjunctions are not the only way statements can be combined. A system which represents both conjunctions and disjunctions, for example, might use a test for $(x = c) \wedge (x' = c')$ which fails even when $B_{xc}(\Upsilon) = 1$ and $B_{x'c'}(\Upsilon) = 1$. For present purposes though, the test for conjunctions of bindings $\bigwedge_i B_{x_i c_i}$ will be defined in terms of individual binding tests by

$$\left[\bigwedge_i B_{x_i c_i}\right](\Upsilon) = \prod_i B_{x_i c_i}(\Upsilon), \tag{2}$$

so every conjunction of represented individual bindings is also represented and, conversely, every representation of a conjunction of bindings also represents the binding in each term. Let tests for conjunctions of bindings and non-bindings be similarly defined:

$$\left[\bigwedge_i B_{x_i c_i} \bigwedge_j \overline{B}_{x_j c_j}\right](\Upsilon) = \prod_i B_{x_i c_i}(\Upsilon) \prod_j (1 - B_{x_j c_j}(\Upsilon)). \tag{3}$$

Note that this makes it impossible for an excursion to represent $(x = c) \wedge (x \neq c)$, because at least one factor in (3) will be 0.

If a representation of $(x = c) \wedge (x' = c')$ automatically also represents $x = c'$ or $x' = c$, then the representation suffers from *crosstalk*. The condition that a variable cannot be bound to

two or more constants prohibits crosstalk. Therefore a test B^*_{xc} for *crosstalk-free* binding of c to x can be constructed from a test which admits crosstalk-afflicted representations and at least one crosstalk-free one, as

$$B^*_{xc}(\Upsilon) = \left[B_{xc} \bigwedge_{c' \neq c} \overline{B}_{xc'} \right] (\Upsilon) = B_{xc}(\Upsilon) \prod_{c' \neq c} (1 - B_{xc'}(\Upsilon)). \tag{4}$$

Similarly, a test for x being unbound is

$$B^*_x(\Upsilon) = \left[\bigwedge_c \overline{B}_x \right] (\Upsilon) = \prod_c (1 - B_{xc}(\Upsilon)). \tag{5}$$

3.2 Inferences

In the representation of inferences it is necessary to propagate variable bindings through rules. Let $P(x^P_1, ..., x^P_n)$ denote the assertion that predicate P is true of variables $\{x^P_1, ..., x^P_n\}$. A rule R states that if P is true of $\{x^P_1, ..., x^P_n\}$, then some predicate Q is true of its variables $\{x^Q_1, ..., x^Q_m\}$,

$$P(x^P_1, ..., x^P_n) \Rightarrow Q(x^Q_1, ..., x^Q_m) \tag{6}$$

and that furthermore some or all of Q's variables are equal to some or all of P's variables; *ie.*, there is a mapping R_V between (subsets of) the rule variables,

$$R_V : \mathbf{x}^P \rightarrow \mathbf{x}^Q. \tag{7}$$

The variables x^P_i and $R_V(x^P_i)$ are equal in the sense that their bindings must agree. A propagation of variable bindings enforces this equality. It must ensure that if x^P_i is bound to c before $P \Rightarrow Q$ is inferred, then afterwards $R_V(x^P_i)$ is bound to c, and no crosstalk is introduced. Therefore, if the time-development operator is to provide a representation of inference in which consequents are automatically represented some time after the antecedents are represented, the bindings and the dynamics together must satisfy:

$$B^*_{x^P_i c}(\Upsilon) = B^*_{R_V(x^P_i)c}(F^\tau \Upsilon) \tag{8}$$

at some time τ for all x^P_i in the domain of R_V.

It is difficult to ensure this condition in all cases, so the examples to be presented will only satisfy the weaker condition

$$B^*_{x^P_i c}(\Upsilon) = B_{R_V(x^P_i)c}(F^\tau \Upsilon). \tag{9}$$

The problem is that two or more inferences may pass different bindings into the same variable of the same predicate. For instance, if rules R and R' give $R(P) = Q$, $R_V(x^P_1) = x^Q_1$, $R'(P') = Q$, $R'_V(x^{P'}_1) = x^Q_1$, then the bindings $x^P_1 = c$ and $x^{P'}_1 = c'$ lead to $x^Q_1 = c$ and $x^Q_1 = c'$. The representations presented here fail in such cases, so the best one can do is to apply tests (4) and (5) before accepting a consequent predicate computed by these systems.

The assertion $P(c_1, ..., c_n)$ is implied by the assertions

'x^P_1 is bound to c_1', ..., 'x^P_n is bound to c_n',
'These are the only bindings of the arguments of P', and
'P with its only argument bindings is true'.

The first set of conditions can be represented by binding tests as usual and the second condition holds by convention, so by introducing a test $P(\Upsilon)$ for the truth of P, $P(c_1, ..., c_n)$ can be represented by:

$$P(c_1, ..., c_n)(\Upsilon) = P(\Upsilon) \prod_{i=1}^{n} B^*_{x_i^{P} c_i}(\Upsilon). \tag{10}$$

Assertions such as $P(x_1, c_2, ..., c_n)$, that P is true for the given bindings and some unspecified value of x_1, are implied by the above statements, omitting those concerning the unbound variables, and can therefore be represented by (10) with the corresponding factors omitted. All the following examples will use this representation.

With representation broken down in this way, inference can be represented by augmenting (8) with

$$P(\Upsilon) = R(P)(F^{\tau}\Upsilon) \tag{11}$$

where $R(P)$ is the test function for the consequent of P under rule R.

3.3 Summary of specification and validation procedure

Within the limitations mentioned earlier, a representation of variable binding and propagation of inferences is given by specifying binding tests $B_{xc}()$, predicate truth tests such as $P()$, and the parameters $\mathbf{w}$ of a time-development operator F^{τ} as explained below. To verify that the representation works it must be shown that any valid combination of variable bindings and predicates can be represented by at least one excursion, and that (11) and (9) hold.

To check that any valid combination of variable bindings is representable, it suffices to check that every possible valid binding of all the variables is representable; *ie.*,

$$\bigcap_{x} B^{*-1}_{xc_x}(1) \neq \emptyset, \tag{12}$$

where x ranges over all variables and c_x is an arbitrary assignment of a constant to x, with the convention that c_x may also be assigned a null value in which case $B^{*-1}_{xc_x}$ is B^{*-1}_{x}. This works because (3) implies that any excursion which represents a particular set of bindings and non-bindings for all the variables also represents all subsets of these.

In all the examples of representations of variable binding given here, the time-development operator F^{τ} is the τ-fold application of the (recurrent) perceptron map

$$y_i(t+1) = \theta(\sum_{j=1}^{N} w_{ij} y_j(t) + w_{i0}). \tag{13}$$

where $\theta(u) = 0$ if $u \leq 0$ and $\theta(u) = 1$ otherwise. In two examples this is extended to a network with time-delayed connections:

$$y_i(t+1) = \theta\left(\sum_{s}\sum_{j=1}^{N} w^s_{ij} y_j(t-s) + w_{i0}\right). \tag{14}$$

An operator F is defined by its weights $\mathbf{w}$. In each example the 'bias' weights are set slightly negative, $w_{i0} = -\frac{1}{2}$, and all weights not otherwise mentioned are set to 0.

4 Examples

Four representations of variable binding and inference are given here. The first two are crosstalk-proof in the sense of (9). The others are not, but require fewer resources and involve more interesting distributed representations.

4.1 Binding grandmother cells

In this simple but wildly inefficient representation, the network has a node for every possible constant-variable pair (x, c), and a node for each predicate P. The test functions are applied to excursions of length 1 (*ie.,* states) $\Upsilon_t = \mathbf{y}(t)$, and simply read out the corresponding node values. Using the same or clearly related symbols to represent corresponding experiences, test functions, and node labels, this is expressed as:

$$B_{xc}(\Upsilon_t) = y_{(xc)}. \tag{15}$$

$$P(\Upsilon_t) = y_P. \tag{16}$$

The weight matrix is specified by

$$w_{R(P)P} = 1 \tag{17}$$

$$w_{(R_V(x_i^P)c)(x_i^P c)} = 1 \tag{18}$$

for all rules R and their variables, together with the conventions mentioned in section 3.3.

Any state with $y_{(xc_x)} = 1$ and $y_{(xc')} = 0$ for $c' \neq c_x$ is in $\bigcap_x B^{*-1}_{xc_x}(1)$. Of these, those with $y_P = 1$ also represent truth of predicate P. (11) and (8) are easily verified.

4.2 Phase binding

This representation, introduced by Shastri [1, 6, 5], uses time to save on space. Nodes y_c, y_x, and y_P are assigned to each constant c, variable x, and predicate P. An excursion of length T is used to represent a binding, with $T \geq V$, the number of variables. The binding test is

$$B_{xc}(\Upsilon_t) = \max_{\tau=t-T+1}^{\tau=t} y_x(\tau) y_c(\tau). \tag{19}$$

In other words, if y_x and y_c ever fire at the same time during the last T time steps, then the binding is represented.

Let δ_{ij} be the Kronecker delta: $\delta_{ij} = 1$ if $i = j$ and 0 otherwise. Let τ_x be a numbering of the variables from 0 to $T - 1$. Any excursion Υ_t such that

$$y_x(\tau) = y_{c_x}(\tau) = \delta_{t-\tau, t-\tau_x} \tag{20}$$

and $y_c(\tau) = 0$ for any remaining constants for all $t - T + 1 \leq \tau \leq t$ is in $\bigcap_x B^{*-1}_{xc_x}(1)$. (If c_x is null then set all $y_x(\tau) = 0$.) Such states survive (4) because only one time step, or *phase* will contribute to (19) for each variable. No constant other than x_c can be 'on' at that time because (20) restricts it to be 'on' in synchrony with some other variable, if ever. Similarly

to grandmother binding cells, setting $y_P(\tau) = 1$ for all τ in the excursion serves to represent truth of predicate P.

To perform inferences, the initial bindings are represented by imposing a temporal pattern $\delta_{t \bmod T,\tau}$ on each participating variable and constant node, using different τ for different bindings. Initially-true predicates are are turned 'on' for all time: $y_P(t) = 1$. The dynamics is defined by $w_{R(P)P} = 1$ and $w^{T-1}_{R_V(x_i^P)x_i^P} = 1$, and works in the obvious way to implement (11) and (8). The time-delayed connections produce phase synchrony between antecedent and consequent variables.

There are further refinements to this system which are not discussed here. In particular, it can be extended to support backwards inferencing for answering queries. Note also that in the restriction $T \geq V$, V can be reduced to the maximum number of variables to be bound during any one related set of inferences.

The grandmother binding-node system represents the $(C+1)^V$ possible binding combinations within the $2^{(C+1)V}$ states of a $((C+1)V)$-node state space. In phase binding only $C+V$ nodes are needed but T time steps are also required, so the size of the excursion space is $2^{(C+V)T}$. This is somewhat larger than the grandmother binding-node state space if $T = V$, but can be much smaller if T is small.

4.3 Tensor-product binding

Smolensky, Dolan, and others have used tensor products to represent bindings [3]. Each constant is associated with an activation pattern over one set of $O(\log C)$ nodes, and similarly for variables. These activation patterns need to have at least one node set to 1. Let γ_i be the index of the $i\underline{\text{th}}$ node used for representing constants, and ν_i be the index of the $i\underline{\text{th}}$ node used for representing variables. Let $Y^c_{\gamma_i}$ be the $\gamma_i\underline{\text{th}}$ component of the pattern assigned to constant c, and define $Y^v_{\nu_i}$ similarly for variable v. Bindings are represented on a set of $O((\log C)(\log V))$ nodes containing a node for each pair (γ_i, ν_j). The test for whether length-1 excursion Υ_t represents that variable x is bound to constant c is

$$B_{xc}(\Upsilon_t) = \prod_i \prod_j \left(y_{(\gamma_i,\nu_j)} \delta_{(Y^x_{\nu_i} Y^c_{\gamma_j})\,1} + \delta_{(Y^x_{\nu_i} Y^c_{\gamma_j})\,0} \right). \tag{21}$$

That is, every binding node (γ_i, ν_j) must be 'on' for which corresponding nodes γ_i and ν_j are 'on' when c and x are represented. (Actually nodes γ_i and ν_j need not physically exist; only nodes (γ_i, ν_j) are necessary.)

Crosstalk is possible in this system; (12) is not guaranteed. However, if relatively few bindings are represented, and the distributed patterns for the constants and variables are sparse, then crosstalk can be shown to be improbable by using reasoning similar to that used in analysing the capacity of direct-product associative memories [7, 2]. These are mathematically similar systems in which the connection weights play the role of the binding nodes of this system.

Inference can be unreliably implemented by a method resembling that used for grandmother cells, replacing (18) with

$$w_{(\nu_i,\gamma_j)(\nu_{i'},\gamma_{j'})} = \sum_R \sum_a \sum_c Y^{x_a^{P_R}}_{\nu_i} Y^c_{\gamma_j} Y^{R_V(x_a^{P_R})}_{\nu_{i'}} Y^c_{\gamma_{j'}} \tag{22}$$

where P_R is the antecedent predicate of R. This elaborate prescription, involving a sum over all rules, their antecedent variables, and constants, ensures that every node involved in representing $B_{RV(x_a^{P_R})\,c}$ is activated by connections from each node involved in representing $B_{x_a^{P_R}\,c}$. Inference can introduce crosstalk, but will tend not to if sparse representations are used.

4.4 Phasic tensor-product binding

A cross between tensor product binding and phase binding is possible. Distributed representations of constants and variables are used as in tensor-product binding, but binding is represented via phase correlations without using extra nodes. Using the node labeling used earlier for tensor-product binding (cf. (19) and (21)),

$$B_{xc}(\Upsilon_t) = \max_{\tau=t-T+1}^{\tau=t} \prod_i \prod_j \left(y_{\gamma_i}(\tau) y_{\nu_j}(\tau) \delta_{(Y^x_{\nu_i} Y^c_{\gamma_j})\,1} + \delta_{(Y^x_{\nu_i} Y^c_{\gamma_j})\,0} \right). \tag{23}$$

Instead of using $(O((\log C)(\log V))$ nodes to represent the direct product at each time step, $O(\log C + \log V)$ nodes are used over T time steps, where $T \leq V$ need be no larger than the number of bindings represented.

Analogously to (20), $\bigcap_x B^{*-1}_{xc_x}(1)$ contains all excursions such that

$$y_{\gamma_i}(\tau) = Y^c_{\gamma_i}(\tau) \quad \text{and} \quad y_{\nu_j}(\tau)\delta_{Y^x_{\nu_j}1} = Y^x_{\nu_j}(\tau)\delta_{Y^x_{\nu_j}1} \tag{24}$$

hold for all i and j for $\tau = \tau_x$, but not for all i when $\tau \neq \tau_x$. That is, different constants must be represented during different phases, ensuring that no variable is bound to two constants. Therefore this system improves on ordinary tensor-product binding in that (12) can be respected.

The wiring for inference is simpler than for ordinary tensor-product binding. (22) is replaced by

$$w^{T-1}_{\nu_i \nu_{i'}} = \sum_R \sum_a Y^{x_a^{P_R}}_{\nu_i} Y^{R_V(x_a^{P_R})}_{\nu_{i'}}. \tag{25}$$

This ensures that if $P(x)$ implies $Q(x^{P_R})$, then both x and x^{P_R} become represented during a common phase. However, all set bits of Y^x and $Y^{x^{P_R}}$ become simultaneously set, so other variables may also become represented as well, introducing crosstalk possibilities. Sparse coding could be used to render this improbable though. If variables with similar significance were similarly coded, then the crosstalk might represent interesting associations which do not follow from the rules.

Another possibility is to prevent crosstalk in either this system or the ordinary phase-binding system by inventing a dynamics F^τ which 'allocates' an unused phase to newly-represented consequent variables.

5 Concluding remarks

A few different representations of variable binding have been illustrated in order to exercise one formal definition. Many other representations are possible, and other variations on the

notion of variable binding might call for modifications to section 3.3. It is hoped that by using an explicit representation of representation itself, via test maps, this type of work can be clarified and consolidated. For example, formalisation of the ordinary phasic and tensor-product representations led naturally to the phasic tensor-product representation, which seems superior to both in resource requirements, crosstalk avoidance, and potential for interesting uses of distributed representations.

Another possible benefit of an explicit representation of representation is automated invention of representations. It is possible to formally state what is required of a representation without explicitly providing one. Therefore particular representations can be sought as solutions to a well-posed constraint satisfaction problem.

There are important issues which are representation-independent. Appropriate explicitness is one. An application of a representation test map requires some computation; how much computation is 'appropriate'? An absurd example shows that some limitations are desirable. In the representation of inference, condition (8) could be dropped if the representation of variable binding extended a test such as (15) with a declaration that an excursion Υ represents binding of c to x if there exists a rule R such that: Its antecedent P is represented, one of P's arguments x_i^P is represented to be bound to c, and $x = R_V(x^P)$. By allowing the representation to draw freely on the predicate calculus, this trick relieves the network of what might naturally have been considered to be its primary task. There appears to be a need for a definition of representational validity which incorporates some notion of 'naturalness'.

References

[1] V. Ajjangadde and L. Shastri. Rules and variables in neural nets. *Neural Computation*, 3:121–134, 1991.

[2] J. T. Buckingham. Delicate nets, faint recollections: A study of partially connected associative network memories. PhD thesis, Edinburgh University, 1991.

[3] C. P. Dolan and P. Smolensky. Tensor product production system: a modular architecture and representation. *Connection Science*, 1:53–68, 1989.

[4] J. B. Pollack. On connectionist models of natural language processing. PhD thesis, Urbana: Computer Science Dept., Univ. of Illinois. (Also Tech. Report. MCCS-87-100, Computing Research Lab., NMSU, Las Cruces, NM.), 1987.

[5] R. Rohwer, B. Grant, and P. R. Limb. Towards a connectionist reasoning system. *British Telecom Technology Journal*, 10:103–109, 1992.

[6] L. Shastri and V. Ajjangadde. From simple associations to systematic reasoning: A connectionist representation of rules, variables, and dynamic bindings using temporal synchrony. *Behavioral and Brain Sciences*, to appear.

[7] D.J. Willshaw, O.P. Buneman, and H.C. Longuet-Higgins. Non-holographic associative memory. *Nature*, 222, 1969.