Hybrid Encoding: The Addressing Problem

Jon M. Slack

Istituto per la Ricerca Scientifica e Tecnologica (I.R.S.T.)
38050 Povo (TN)
ITALY
e-mail: slack@irst.it

Abstract

Locality constraints are generally assumed to have their source at the hardware, or neuronal, level. However, the paper shows that the way symbols address constituent structure represented at the connectionist level limits their access to the encoded information. These limitations are expressed as the constructs of local and address domain and provide an explanatory basis for a wide range of cognitive constraints.

Introduction

Cognitive Scientists have recently turned their attention to the exploration of hybrid symbolic-connectionist architectures (Hendler 1991). Fodor and Pylyshyn (1988) have claimed that the explanation of mental functioning is properly located at the symbolic level and that connectionism provides little more than an implementation paradigm for such models, making issues of hybrid architecture irrelevant. This claim is refuted by showing that the hybrid nature of cognitive architecture is itself the source of constraints which have a powerful influence on shaping mental abilities.

In outline, the structure of the argument to be presented is as follows: A number of researchers have shown that constituent structure can be encoded in terms of connectionist representations and combinatorial operators (see Hinton 1990; Pollack 1990; Touretzky 1990; Smolensky 1990). Fodor and Pylyshyn (1988) view this as a demonstration that Connectionism merely provides an implementation paradigm for the symbolic-level explanation of cognition. But even as an implementation it is necessary to specify how symbols make contact with the underlying connectionist encodings. This is a problem because, as Fodor and McLaughlin (1990) argue, such encodings are not directly accessible for symbolic addressing. Instead, it is proposed that symbolic access is only possible through the use of a particular communication protocol, one based on the general view of structural configurations as comprising an address structure and a set of constituents. This particular protocol is forced on hybrid systems by the nature of information representation at the connectionist level. The next section outlines these properties.

Generally, locality constraints are explained in terms of hardware limitations or neuronal properties. An alternative explanation can be cast in terms of a restrictive communication protocol mediating between the levels of an hybrid symbolic-connectionist architecture. The third section outlines the form of this communication protocol.

The basic thesis being proposed is that certain forms of locality imported into symbolic accounts of cognition derive from the fact that symbolic structures are encoded in a medium that imposes natural local restrictions. In the fourth section, these constraints are derived from the symbolic addressing protocol. Finally, it is suggested that such constraints might underlie a wide range of locality properties originally invoked to account for the observed features of human cognition.

E-Space: Encoding Space

The question of the most appropriate assignment of cognitive phenomena to symbolic and connectionist levels of explanation is left as open. What is being proposed is that the mapping between the two levels provides an appropriate basis for the explanation of locality constraints, even when only minimal assumptions are made concerning the nature of the relationship between them.

Encoding structure in an hybrid symbolic-connectionist architecture involves showing how the notion of combinatorial structure can be preserved in passing from the symbolic level to an encoding medium comprising points in a vector space. Slack (1990) shows how the operations of association and superposition can be used to define a connectionist representation medium, referred to as *Encoding Space*, E-space, consisting of composed connectionist states. Formally, the structure of E-space can be defined as a *semiring*, as follows:

Definition E-space comprises the quintuple $(V,+,**,\mathbf{0},\partial)$
1. $(V,+,\mathbf{0})$ is a commutative monoid defining the superposition operator;
2. $(V,**,\partial)$ is a monoid defining the association operator;
3. $**$ distributes over $+$.

V is the set of points, or corners, of a vector space, and the two identity elements, $\mathbf{0}$ and ∂, correspond to identity vectors for the operations of superposition and association, respectively. It is important to note that the set V is closed under both operations.

We can now define a representational mapping, referred to as the *encoding homomorphism*, f_e, which preserves combinatorial structure in mapping the set of symbolic

languages L_X, that can be defined over an alphabet of symbols, X, using the combinatory operations of concatenation, denoted by '.', and union, denoted by '$\cup$', into E-space. The homomorphism is defined as follows:

Definition The encoding homomorphism, f_e, maps the semiring $(L_X, \cup, ., \emptyset, \{\varepsilon\})$ into the semiring $(V, +, **, \mathbf{0}, \partial)$, such that

$$f_e(x \cup y) = f_e(x) + f_e(y)$$

$$f_e(x.y) = f_e(x) ** f_e(y), \text{ where } x, y \in L_X.$$

Thus, through f_e the set of languages that can be defined on a symbolic alphabet can be encoded in E-space, preserving their combinatorial structure.

This representational framework establishes a formalism for describing the composition of distributed representations which can encode constituent structure. It is not intended as a specific connectionist representation scheme in that it does not specify the details of particular connectionist operators. Rather, it captures the general properties of an encoding medium comprising a vector space and the operations of association and superposition.

Addressing Encoded Structures

The encoding mapping, f_e, can be used to build composite E-space vectors but it isn't clear how the encoded information can be accessed. For example, consider the structured vector built using the operations of association and superposition whose composition is described by the expression given in (1)

$$[\mathbf{a}**[\mathbf{c}**w + \mathbf{d}**x] + \mathbf{b}**[\mathbf{f}**y + \mathbf{g}**z]] \ldots (1)$$

where the bold letters denote elements of the set **V**. This symbolic expression describes the vector composition tree shown in figure 1.

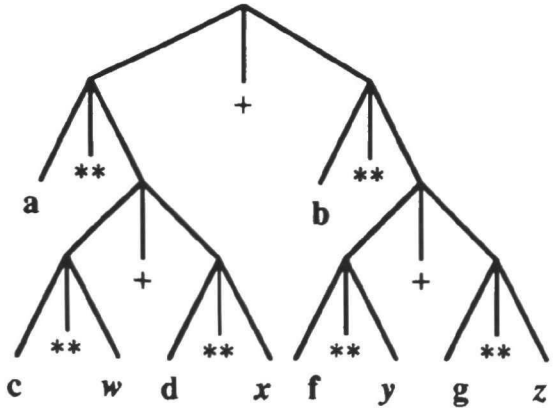

Figure 1. Vector Composition Tree

As an E-space encoding this composite structure is realised as just a point in vector space, its implicit structure is not apparent in its encoding. This problem is at the heart of Fodor and McLaughlin's (1990) argument against connectionist structure. As Fodor and McLaughlin put it,

> ...when a complex Classical symbol is tokened, its constituents are tokened. When a tensor product vector or superposition vector is tokened, its components are not .....the components of tensor product and superposition vectors can have no causal status as such.
> (Fodor & McLaughlin 1990, p. 198)

This is illustrated by the complex symbol [[John] [gave [Mary] [the ball]]] where each component is tokened as part of the composed structure's token. With composed vectors, on the other hand, the constituents are not directly accessible. The problem is to specify how the constituents of a composed E-space encoding can be accessed such that mental processes can be sensitive to their structure.

In an hybrid architecture mental processes can be defined at two levels; symbolic processes and connectionist processes. The latter class of processes might operate on compound vector encodings exhibiting a form of *systematicity* equivalent to that underlying symbolic processes and structures, as suggested by van Gelder,

> A second and more radical approach is to devise models in which structure-sensitive processes operate on the compound representations themselves without first stopping to extract the basic constituents. These processes must capitalize directly on the inherent and systematic structural similarities among nonconcatenative representations. In such models it is not only storage that takes place in the nonconcatenative domain, but the primary processing responsible for systematic cognitive behavior as well.
> (van Gelder 1990, p. 381)

However, the potential for such processes remains to be explored and the focus of this paper is on the problem of how structures encoded in E-space can be made accessible to processes defined at the symboilic level. Some view this as a trivial problem of implementation (Fodor and McLaughlin, 1990; van Gelder, 1990), but it will be shown that important cognitive constraints have their origins in this problem of access. To solve this problem it is necessary to show how symbols make contact with, or address, the nodes of vector composition trees. The problem is viewed as one of defining an address system for composed E-space encodings.

With complex symbols such as [[John] [gave [Mary] [the ball]]] the constituents are made explicit by the parentheses; each pair of parentheses denotes a non-terminal node in the symbolic derivation tree. When Fodor and Pylyshyn (1988) speak of mental processes being *sensitive* to the structure of such symbols, they mean that the processes can access the individual constituents on the basis of their structural configuration. The configuration of the symbol is given by the embedding of the parentheses, which expresses the derivation of the symbol string. The problem with E-space encodings is that the equivalent configuration is not directly accessible to symbolic-level processes. What we need to develop is a general concept of *address structure* which specifies how information encoded in a structure can be accessed, regardless of whether it is encoded symbolically or as an E-space vector.

In general, symbolic structures support two basic forms of addressing, *direct addressing* and *relative addressing*. The former involves direct access to constituents. For example, in the complex symbol the pairs of parentheses distinguish

the different components and can be used to access them directly provided they are individuated through some form of indexing or labeling scheme. This is equivalent to uniquely labeling each nonterminal node of the symbol's derivation tree. In addition, the individual words label the terminal nodes of the tree. The parentheses can also be used to define a relative addressing system in terms of their depth of embedding and/or sequential order. For example, the component *Mary* can be located using the address *the first set of paretheses at the second level of embedding*. Such an addressing scheme requires (i) a pre-defined origin to function as a 'zero' address, and (ii) a set of labels for distinguishing the elements of the path connecting the origin to the addressed component. From these examples, it is clear that the address structure comprises a structural basis, the symbol's derivation tree, plus some form of addressing scheme, that is, a set of labels over which a formal language can be defined, for accessing the individual constituents. Whereas direct addressing is insensitive to the structural configuration of the symbol, relative addressing schemes utilise it to encode addresses. Thus, a direct addressing scheme merely comprises a set of labels plus a mapping taking elements from this set onto the set of constituents. A relative addressing scheme, on the other hand, requires both a set of labels and a structural configuration over which to define paths, plus a mapping which assigns the labels to elements of the configuration. Both addressing schemes are necessary for symbols to access constituent structure encoded in E-space.

Dual Addressing

E-space encodings require a direct symbolic addressing scheme so that the encoded constituents can be processed as 'chunks'. This type of distal access (Newell 1980) which is the capacity to address a structure in some remote, abbreviated manner, is fundamental to any notion of compositionality as without it a structure could only be referenced through its full specification. In a classical complex symbol, the parentheses function to indicate the 'chunking' of the symbol into constituents. However, each chunk needs to be uniquely labeled for it to be accessed directly. This reference capability also instantiates *the principle of explicit naming* (Marr 1982). According to this principle if you want to refer to a 'thing' so that you can describe 'it' or reason about 'it', then begin by giving 'it' a label. Once an entity has a label, it can be used in repeated applications of structure building operations to derive larger entities, which themselves require labeling if they are to be processed as single entities.

Relative addressing schemes are important for expressing the address of one constituent relative to another. This referential capacity is crucial to the notion of constituent structure as it captures the mother-daughter, or dominance, relationship expressed in composition trees. 'Daughter' nodes can be addressed relative to their 'mother' nodes. This capacity is utilised in building composite structures where a constituent can only be identified relative to an existing node in the structure. In such circumstances only a relative addressing scheme is available.

Symbolic access to E-space encodings needs to support both of these referential capacities and thus, an address structure incorporating both direct and relative addressing schemes is required.

Address Structure DAGs

Symbolic addresses access E-space encodings through the operation of *retrieval*. This operation delimits the communication protocol between the levels of the hybrid architecture. In most connectionist systems retrieval is the inverse operation of association. We can define a general retrieval operator, denoted #, as follows: Given the composite vector, $\mathbf{v_1} = [\mathbf{a**b} + \mathbf{c**d}]$, then

$$\mathbf{a} \# \mathbf{v_1} = \mathbf{b} \quad \ldots (2)$$

Because the association operator is noncommutative we can distinguish between its arguments, similar to the role-function distinction maintained in Smolensky's representation scheme (1990). The arguments are labeled *address* and *constituent* according to the interpretation given in (3),

[address vector] ** [constituent vector] . (3)

Within this protocol, **a** is interpreted as the address of the constituent vector, **b**, within the composite encoding, $\mathbf{v_1}$. Thus, when the composite vector is accessed using the address vector, **a**, it retrieves the constituent vector, **b**.

This address-constituent protocol is independent of the form of the address and thus supports both direct and relative addressing. Direct addressing is achieved by associating each constituent vector of an E-space encoding with an address vector corresponding to a symbolic label. This label can then retrieve the constituent vector through its assigned address vector. Symbolic labels that function in this way are referred to as *global labels*.

Relative addressing is the natural mode of operation of the retrieval operator in the sense that an address vector specifies the address of a constituent vector relative to a particular composite encoding. For example, **a** is the address of **b** relative to $\mathbf{v_1}$, but would not retrieve **b** given a different composite encoding. In this sense, global labels can also be regarded as defining relative addresses; they specify the direct address of constituents relative to the top-level composite encoding, that is, the vector that encodes the complete structure. Symbolic labels assigned to address vectors that retrieve constituents relative to other constituent vectors are referred to as *local labels*.

Via the retrieval operator, the constituents of an E-space encoding can be accessed using both direct and relative symbolic addresses. The full set of such addresses comprises the *address structure* of the composite encoding. Because symbolic addressing is mediated through the address-constituent protocol these structures are restricted to the class of *regular languages*. This allows address structures to be represented as directed acyclic graphs (DAGs), in which the edges and nodes are assigned local

labels and global labels, respectively. Figure 2 shows the address structure DAG (AS-DAG) for the compound vector given in (1) and whose E-space encoding is shown in figure 1. The local labels are denoted by the lower-case letters, a-g, and the global labels by the upper-case letters, A-C. The global labels function as direct addresses for the constituent vectors within the composite E-space encoding. The local labels, on the other hand, specify the address of a constituent relative to its superordinate ('mother') constituent.

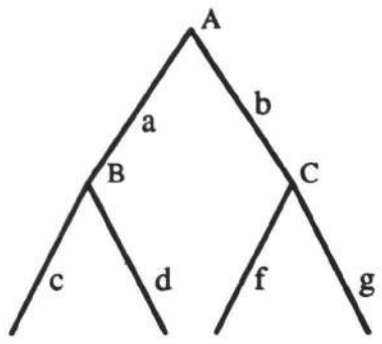

Figure 2. Address Structure Tree (AS-DAG)

In fact, the AS-DAG shown in figure 2 represents the union of two separate address structures both of which take the global label *A* as their root-node. It is important to remember that vector composition trees, such as that shown in figure 1, specify the encoding of composed vectors, but are not necessarily accessible at the symbolic level as connected configurations. An AS-DAG corresponds to a symbolic level 'view' of a composite E-space encoding. However, because they have a restricted form, limited to the class of regular languages, the full address structure for a particular E-space encoding may comprise the union of a set of AS-DAGs, each providing a view of a different part of the composite configuration.

There remains the problem of how the labels of an AS-DAG are assigned vector values under the encoding mapping. Because AS-DAGs incorporate a dual-addressing system the resulting redundancy requires the dynamic assignment of vector values to at least one of the two sets of labels. Local labels capture local structure and the assignment should therefore be both *faithful* and *consistent*, that is, elements of the set of local labels, denoted by $\mathcal{L}$, should map to different elements of $\mathcal{V}$, the set of vectors, and the mapping should be constant, at least local to a given address structure. Global labels are unique relative to a given address structure by definition and so the problem of consistency does not arise. However, the mapping should be *faithful* (elements of the set of global labels, denoted by $\mathcal{G}$, should map to different elements of $\mathcal{V}$)[1].

[1] As defined, the sole function of global labels is to retrieve constituents directly from composite E-space encodings and as such the relationship between their vector values and the constituent vectors they access can be quite arbitrary. There is no necessity for the address labels to comprise partial descriptions of their associated constituents as implemented in some hybrid architectures (Touretzky 1990). This independence allows the vector values of global labels to be determined as a function of the AS-DAG configuration rather than being related to the constituent vectors they access.

These constraints give a higher priority to fixing the vector values of local labels under f_e. In this case, the values of the global labels are determined by (i) assigning a vector value to the root-node of the AS-DAG, and (ii) propogating the vector values down the tree through association with the fixed local label vector values. Thus, an AS-DAG functions both as the component of f_e that maps global labels onto vector values, and as the symbolic 'view' of the local structural relations existing between the constituents the global labels reference.

In summary, hybrid encoding captures combinatorial structure through the association of a composite E-space encoding with a corresponding address structure, the latter specifying the symbolic-level access to constituent encodings. If the address structure is well-formed, then the E-space encoding has a unique decomposition.

An Example Address Structure

If the idea of hybrid encoding is valid then symbols functioning as global and local labels should be implicit in Classical symbolic cognitive representations. In particular, linguistic representations should embody such functional labels as language is the prime example of a symbolic address system. To illustrate the use of these labels in the

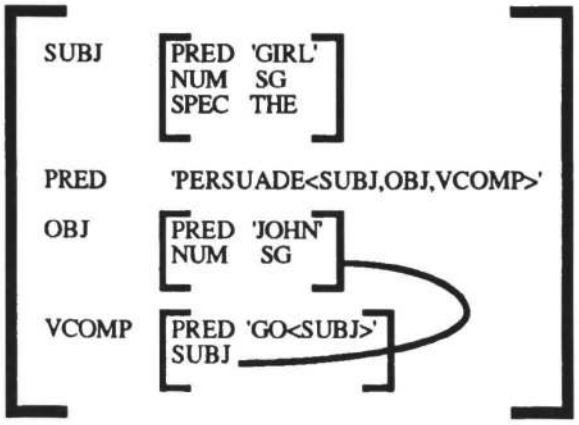

Figure 3. LFG representation of the sentence "The girl persuades John to go"

mental representation of syntactic structure, consider the

LFG[2] representation of the sentence *The girl persuades John to go* shown in figure 3. In this structure, each pair of parentheses signifies a functional structure (f-structure), thereby designating a constituent of the overall structure. The grammatical functions, SUBJ, OBJ, etc., label local structure in that they specify the relationships between constituent f-structures. Such symbols function as local labels and in the hybrid encoding framework are assigned to AS-DAG edges. This is shown in figure 4 which depicts the AS-DAG for the E-space encoding of the f-structure shown in figure 3. Thus, AS-DAG edges designate local symbolic structure, that is, the structural relations holding between adjacent constituents.

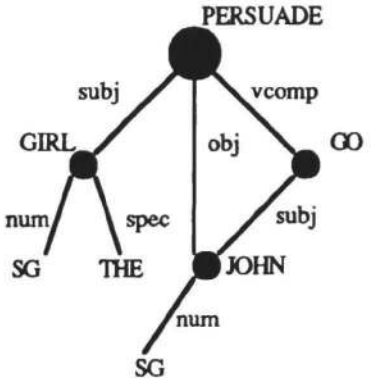

Figure 4. AS-DAG for the LFG representation in figure 3

The AS-DAG edges leading to terminal nodes represent a different form of local structure corresponding to basic featural relations, such as, *number* or *specification*, and are assigned feature labels (num, spec, etc.). The terminal nodes of the AS-DAG correspond to the values of the basic features, for instance, SG (singular) for the feature *number*, and so on. Both feature labels and function as local labels and thus, have fixed E-space encodings.

Global labels correspond to any set of symbols within a representation that function to distinguish the individual structural constituents. In figure 3, the parentheses denote the constituent f-structures, but they are themselves undifferentiated as labels. However, associated with each set of parentheses is a predicate label which uniquely identifies each component f-structure and these function as global labels, being assigned to the AS-DAG nodes in figure 4.

[2] LFG, or Lexical Functional Grammar, is a syntactic formalism which encodes predicate-argument structures in terms of the grammatical functions specified in the sentence. For more details see, for example, Kaplan and Bresnan (1982).

Locality Constraints on AS-DAGs

A number of important locality constraints emergent from the properties of hybrid encoding can be defined on AS-DAGs. These constraints provide the basis for defining the constructs of *local* and *address domain*.

As AS-DAG representations are drawn from the class of regular languages they can be generated by right-linear grammars where each production has the form,

$$A \rightarrow \omega B \quad \text{or} \quad A \rightarrow \omega$$

and where $A, B \in \mathcal{G}$, $\omega \in L_{\mathcal{F}}$, and $\mathcal{F}$ is the set of labels which have faithful and consistent encodings under f_e. This allows each level of an AS-DAG structure to be defined as a domain local to a particular global label.

Complex relative addresses can be defined on AS-DAGs as strings comprising a global label, functioning as the origin node, followed by a string of local labels of arbitrary length. For example, given figure 4, we can define the address PERSUADE-vcomp-subj-num to reach the terminal node with the value SG. However, this structural referential capacity is in fact restricted to *nearest neighbour addressing*. That is, AS-DAG nodes can only be addressed relative to their mother node, and relative addresses defined on AS-DAGs comprise the global label of the mother node plus at most one local label (that is, one AS-DAG edge). This is because structural AS-DAG addresses connect points in E-space and between any two such points there are an infinite number of possible paths linking them, but only the minimal path is distinguishable. Thus, unless the intermediate nodes of an E-space path are differentiated through labeling, that is, direct addressing, the path is not recoverable, only the end point is accessible.

These different constraints can be integrated into a single construct, referred to as a local domain, incorporating the following properties:

(a) A local domain is indexed by a unique **global label**;
(b) associated with each global label is a **finite local structure** comprising local labels (AS-DAG edges) which have vector values fixed relative to the value of the global label; and in addition,
(c) a local domain can dominate upto **one** subordinate local domain.

The full definition is as follows,

Definition A local domain is a labeled AS-DAG comprising a global label at its root node, a fixed local structure, that is, a finite set of descending edges, such that each edge is assigned a local label, with a maximum of one such label addressing a non-terminal AS-DAG node.

In fact, we can distinguish between terminal and non-terminal local domains, with only the latter having property (c). At the intuitive level, local domains can be thought of as the permanent building blocks used to compose structured E-space encodings. All the information encoded in a local domain is accessible from its global label, as the vector value of each daughter node is known relative to the root node.

Address domain.

Address structures are built up through the composition of

local domains. This operation is schematized in figure 5 in which local domains are represented as productions. In this figure, the domain associated with the global label A_1 specifies the address of a local domain, X, the identity of which is 'unknown' prior to composition. In composing the next level of structure, this address is attached to the new local domain labeled B_1 through the process of *address unification*. In assigning the address of X to B_1, the relative addresses encoded in its local domain become instantiated as E-space encodings. This is because the domain contains local labels which specify E-space addresses relative to the vector value of the domain's global label. Once the global label receives a value, the relative addresses are automatically established.

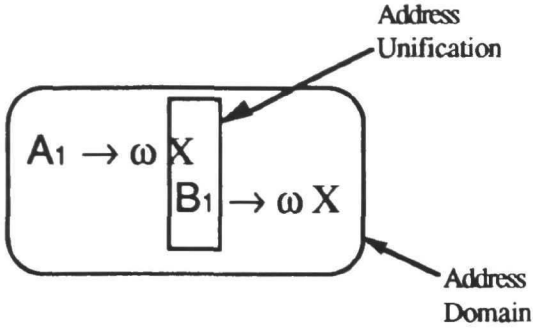

Figure 5. Address domain for A1

The operation of composition can be viewed from the opposite direction and seen as linking the contents of the constituent B_1 to the address specified by X. In this sense the composition operator instantiates the address-constituent protocol. Hence the protocol spans adjacent local domains. In figure 5, the address part of the protocol is specified relative to A_1 and the constituent part is the local domain B_1. Thus, the protocol through which symbolic processes access E-space encodings can be defined on AS-DAGs as a window spanning two adjacent local domains, indexed to the global label of the most dominant of the two domains. This construct of a symbolic addressing window is referred to as an *address domain* and is defined as follows:

Definition The address domain indexed to an AS-DAG node, A, comprises the local domain of A plus the subordinate adjacent local domain attached through composition.

To illustrate the importance of this construct, consider the representation shown in figure 3, where the OBJ *f*-structure is co-indexed with the *f*-structure of the (VCOMP SUBJ) function, as indicated by the link. In formalisms such as LFG, these indexations are expressed as control equations of the form

(vcomp subj) = (obj) . . . (C1)

specified as part of the lexical entry for the predicate (in this instance, the predicate PERSUADE). Assuming that LFG predicates have the properties of global labels, then the AS-DAG address referred to by the left-hand term of the equation is within the address domain of the predicate PERSUADE, and hence, accessible for symbolic indexing. However, such an address cannot contain more than two local labels as this would overflow the address domain that limits symbolic level access This explains why no known natural language requires control equations of the form

(vcomp vcomp subj) = (obj) . . . (C2)

where the left-hand term involves control across three levels of nested structure. In other words, the control equations attached to a predicate can only be defined within its address domain. While the LFG framework can capture such constraints it cannot explain them in any principled way. The present framework, however, provides a natural basis for such a constraint.

Conclusions

Elsewhere the construct of a local domain has been used to explain unbounded dependency phenomena (Slack 1990). In the longer version of this paper (Slack 1991) a wide range of locality constraints inherent in natural language phenomena are accounted for in terms of the functions of local and global labels. However, the full range of applicability of these ideas remains to be explored.

References

Fodor, J.A., and McLaughlin. B. P. 1990. Connectionism and the problem of systematicity : Why Smolensky's solution doesn't work. *Cognition* **35**: 183-204.

Fodor, J.A., and Pylyshyn. Z.W. 1988. Connectionism and cognitive architecture: A critical analysis. *Cognition* **28**: 3-71.

Hendler, J. 1991. Special issue of *Connectionist Science* on hybrid symbolic-connectionist architectures. Forthcoming.

Hinton, G.E. 1990. Mapping part-whole hierarchies into connectionist networks. *Artificial Intelligence* **46**: 47-77.

Kaplan, R.M. and Bresnan, J. 1982. Lexical-Functional Grammar: A formal system for grammatical representation. In J. Bresnan (ed.), *The Mental Representation of Grammatical Relations*. Cambridge: MIT Press.

Marr, D. 1982. *Vision*. San Fransisco:Freeman.

Pollack, J.B. 1990. Recursive distributed representations. *Artificial Intelligence* **46**: 77-107.

Slack, J.M. 1990. Unbounded Dependency: Tying Strings to Rings. In *Proceedings of COLING-90*, Helsinki, Finland.

Slack, J.M. 1991. Hybrid Encoding as a Source of Cognitive Constraints. Forthcoming.

Smolensky, P. 1990. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence* **46**: 159-217.

Touretzky, D.S. 1990. BoltzCONS: Dynamic symbol structures in a connectionist network. *Artificial Intelligence* **46**: 5-47.