THE RIGHT OF FREE ASSOCIATION:
Relative-Position Encoding for Connectionist Data Structures

John A. Barnden

Computing Research Laboratory
New Mexico State University

DATA STRUCTURING IN CONNECTIONIST SYSTEMS

The challenge presented to connectionism by high-level cognitive processing — which includes reasoning, planning, and some aspects of natural language understanding — is gaining increasing recognition, and attempts to meet it are becoming more common. The main technical difficulties are listed in Barnden (1983, 1984, 1986, 1988a), McDermott (1986) and Norman (1986). They include the well-known variable-binding problem and the problem of accounting for complex, temporary, novel data structures. An example of such a data structure is an internal rendering of the information in the sentence "John believes that Pat gets angry whenever Tom talks about going to Tibet".

The thesis of this paper is that a promising way to approach the mentioned challenge is to encode complex temporary data structures by means of two somewhat atypical techniques, which I call "Relative-Position Encoding"(RPE) and "Pattern-Similarity Association" (PSA). These are answers to the following question, which lies at the heart of the technical difficulties mentioned above:

> **The Temporary-Association Question**
> *How are pieces of information put into temporary association with each other in a connectionist system?* (For instance, how are the different parts of a complex temporary proposition put together? How are variables bound to values?)

A variety of connectionist techniques have been proposed in answer to this. It is hard to categorize them satisfactorily, but the following rough classification will suffice for present purposes:

- *weight-change techniques*
- *binder techniques*
- *pattern-relationship techniques* (including PSA)
- *positional techniques* (including RPE).

These classes should be thought of as possibly-overlapping regions within a complex space of possibilities, rather than as being rigidly delineated.

Weight-change techniques involve temporary weight changes on connection paths between nodes or subnetworks representing the information items to be put into association. An (excessively) simple example would be a temporary weight increase on a single connection between two network nodes, of which one represents the idea of being hungry and the other a particular person, John: the intent being to temporarily represent the proposition that John is hungry. More complex examples would involve weight changes on groups of connections or connection chains joining large and possibly distributed sets of nodes. Important recent examples of weight-change techniques are provided by systems using multiplicative connections [Pollack 1987] or programmable networks [McClelland 1986].

The simpler forms of binder technique involve a temporary activation change on a "binder" node dedicated to the particular combination of nodes or subnetworks representing the information items to be put into association. More complex forms of binder technique allow the binder to be a subnetwork rather than a single node, and/or allow different activation states of the binder to indicate different bindings. Smolensky (1987) has provided a general framework encompassing binders. Binders appear in the systems of Cottrell (1985), Derthick (1987), Touretzky & Hinton (1985), and others.

The binder is usually connected in a straightforward way to the nodes/subnetworks it can bind, and in that case an unusual activation level on the binder can be construed as a temporary marking of a connection chain joining those nodes (cf. Feldman's (1982) dynamic connections). The binder technique is

thereby closely related to weight-change techniques.

In pattern-relationship techniques, the temporary association consists of the exploitation of some given relation on activation patterns. An example is provided by the "reduced descriptions" technique [Hinton 1987, Touretzky & Geva 1987]. However, what is more relevant to the present paper is a special type of pattern-relationship technique, called "*Pattern-Similarity Association (PSA)*" techniques. A (over)simple example of the idea is as follows. Imagine two subnetworks h and j, each divided into two parts called the Information Part and the Associator Part. The temporary proposition that John is hungry could be encoded by letting the Information Parts of j and h contain activity patterns representing John and the idea of being hungry, respectively, and letting the Associator Parts of h and j contain the same activity pattern X. The subnetworks h and j are deemed to be temporarily associated by virtue of this equality of their "associator" patterns (X). In more sophisticated versions of the idea the required similarity of patterns need not be equality, and, in principle at least, there need be no separate Associator subnetworks — associator patterns could be mixed in with information patterns.

There is a sense in which PSA techniques are really binder techniques of an advanced sort. In the simple example above, if several Associator Parts contain the same activity pattern X, then the union of these subnetworks can be construed as a binder containing a specific activation pattern indicating the binding together of the subnetworks. The binders thus defined, being unions of Associator Parts, are highly distributed, overlapping subnetworks. They are so different from the binders usually entertained in connectionist models that it is worth isolating PSA techniques as a special class.

Limited forms of positional technique crop up in many connectionist systems, especially those concerned with visual perception. The idea is best approached by means of a simple prototypical example. Consider a word-perception system that has a separate register-like subnetwork for each letter position in words. A particular pattern of activation distributed over the ith subnetwork would represent the presence of a particular letter at position i. Implicitly, therefore, the patterns of activation in the subnetworks encode *temporary associations* — relative positions — of letters in a word. Thus, the technique is termed "positional" because temporary association is achieved *merely by placing suitable activation patterns, encoding the information items to be associated, in suitable "positions" in the total network.* There is no weight change, activation of binders, or use of special associator patterns. (The name of the positional technique does *not* derive from the fact that what is encod*ed* in many applications is position in words or some other type of space.)

The (quasi-)connectionist sentence-parsing system of Charniak & Santos (1987) uses a more interesting and advanced positional technique that is highly germane to this paper. The model is centered on a two dimensional array of registers, which could in principle be implemented as connectionist subnetworks. By putting the registers into suitable states representing syntactical categories, the array can hold parse trees. The tree structure is encoded to a significant extent by the relative positioning of states in registers. We therefore call the technique a form of "*Relative-Position Encoding (RPE)*".

The working hypothesis that, to address the abovementioned "challenge" adequately, what is needed is an advanced form of RPE, has underlain my own development of connectionist models [Barnden 1985, 1986, 1987, 1988a,b], which has proceeded independently from the work of Charniak and Santos. The next section will sketch my present model, called "Conposit", which relies on an RPE technique broadly similar to that of Charniak & Santos, but more general and thorough-going than theirs. Conposit makes crucial use, too, of a PSA technique.

SKETCH OF CONPOSIT

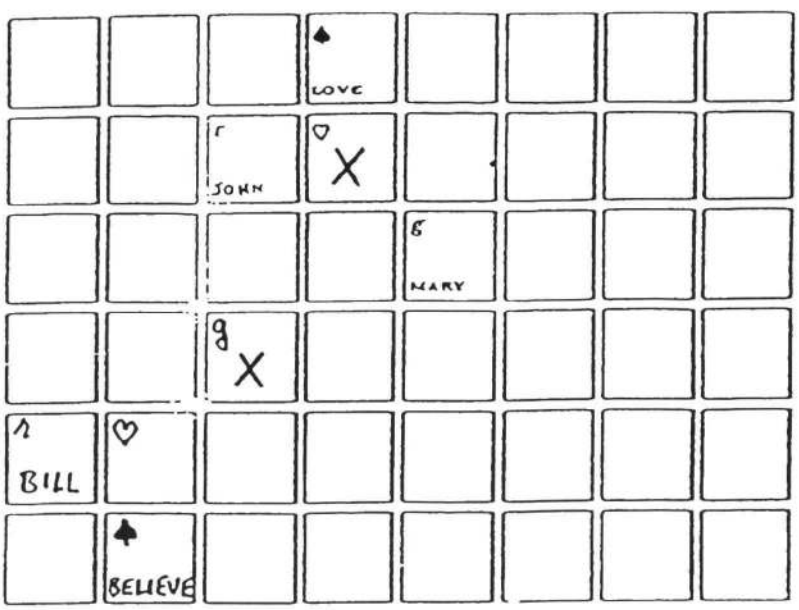
This section is of necessity highly simplified, and concentrates on Conposit's representational tools rather than its processing methods. Representation and processing details can be found in Barnden (1986) supplemented by Barnden (1987, 1988a,b).

Conposit is currently only "quasi-connectionist", in the sense of being a computational architecture whose components can be straightforwardly implemented in connectionist terms [see Barnden 1986 for some suggestions]. A detailed connectionist implementation will be simulated in future research. The two-stage strategy adopted — mapping information processing to an intermediate-level model, and later mapping this to the detailed connectionist level — is done in a spirit analogous to good design practice in computer science. The intermediate level has all along been constrained by the need for later connectionist implementation.

The versions of Conposit simulated to date are centered on a 32×32 array of registers, called the "configuration matrix (CM)". (The small, 2D nature of the CM is motivated by the fact that Conposit is intended to be a rough, preliminary model of high-level processing in brain cortex, and the CM is taken to be implemented as a small region of the cortical sheet. Individual registers are taken to be implemented as cortical columns.) The purpose of the CM is to hold *temporary* data structures being manipulated in reasoning and other high-level cognitive tasks. Conposit as currently simulated has no long-term memory for data structures, but see Barnden (1986) for an LTM proposal.

The content of the CM at any moment is given by a function σ from the CM registers to a state set B. Each member of B is an ordered pair (s, h), where s is a "symbol" and h is a vector of ON/OFF values for a fixed set of "highlighting flags". Both s and h for a given register in the CM are assumed to be implemented as activation patterns over part of the small neural/connectionist subnetwork forming the implementation of the register. Any symbol can be placed in any register. A symbol may have a specific representational function, such as denoting a particular person or a particular type of relationship among people. The significance of highlighting will become apparent.

The associations between the parts of a temporary data structure are encoded by (a) *adjacency relationships among values in CM registers,* often supplemented by (b) *sharing of symbols by several registers.* The RPE and PSA aspects of Conposit reside in (a) and (b) respectively. The encoding is exemplified by the Figure below, which shows a possible state of a region within the CM.



This region contains a representation of the proposition that *Bill believes that John loves Mary.* Each square shows a register. The words and capital letter X indiacte specific symbols s. Absence of symbol word indicates an occurrence of a special "null" symbol. The spade, heart, 'r' and 'g' symbols indicate ONness of four specific highlighting flags, called black, white, red and green respectively. The

symbol LOVE permanently denotes the class of all possible situations in which one person loves another. Any register containing this symbol also temporarily denotes that class. Any white-highlighted register adjacent to a LOVE-containing, black-highlighted register is deemed to denote, temporarily, a *member* of the class — that is, a loving situation. Such a register appears near the middle of the Figure. Any red-highlighted register adjacent to a register denoting a loving situation is deemed to denote the agent of the loving; similarly for green highlighting and the object of the loving. Thus, the upper "subconfiguration" of register states shown in the Figure encodes the proposition that John loves Mary.

So far, then, temporary association is a matter of exploiting the permanent, 2D structure of the CM, by having symbol occurrences and highlighting in suitable relative positions. In fact, the encoding relies only on the undirected *adjacency* relationships in the CM. Ways of using the CM that make broader appeal to the permanent structure in the CM — notably by using the CM as a map of some region of space in certain simple types of spatial reasoning — are discussed elsewhere [Barnden 1986, 1987].

The symbol X in the Figure is one of a special class of "unassigned" symbols, that do not permanenty denote anything and are akin to variables in a logic. However, by virtue of its position in the head register (i.e. the white-highlighted one) in the John-loves-Mary subconfiguration in the Figure, X is deemed to *temporarily* denote the loving situation denoted by that register. But X also appears in another register in the Figure, and makes that register also denote the loving situation. Thus, symbol-sharing is being used to temporarily associate registers in the sense of making them temporarily denote the same thing. This is an instance of Conposit's use of PSA (Pattern-Similarity Association). Because the lower X-containing register in the Figure denotes the situation of John loving Mary, the lower subconfiguration in the Figure encodes the top level of the proposition that Bill believes that John loves Mary. (PSA can also be used to split up propositions like John-loves-Mary into several pieces.)

The processing of the short-term data structures in the CM is performed by internal and external "circuitry" — system components that will be mapped straightforwardly into a connectionist implementation. The internal circuitry mediates mainly neighbor-neighbor interaction among registers in the CM, whereas the external circuitry embodies hardwired *condition-action processing rules*. Rules can detect particular configurations of symbols and highlighting states in the CM, by means of highly parallel detection circuitry that involves further two-dimensional register arrays called *location matrices* (see below). The response of a rule is to send a complex sequence of "CM signals" to the CM. The generation of the sequence can involve conditionals (testing the CM state), loops, and a simple form of non-recursive routine calling. A CM signal affects the CM in a highly SIMD-like, register-local, parallel fashion: the signal is distributed identically to each CM register, whereupon different registers change state differently, according to their own current states and those of their immediate neighbors.

A CM signal can have one of a number of effects, such as: changing the states of some highlighting flags in each register that is highlighted in some specified way and that has at least one neighbor highlighted in some other specified way. It is also possible for a signal only to have an effect on a randomly chosen register satisfying the highlighting conditions, rather than on each such register. Barnden [1986] details how the signals can be used to process data structures, and, in particular, to find free space for, and then create, new data structures in CMs. Specific information processing tasks are reported on in the next section.

Although there is no space here for a detailed account, there is a sense in which the binder technique for temporary association is used behind the scenes. The binders are the elements of some of the "location matrices"(LMs) alluded to earlier. An LM is a 32×32 matrix of registers, each containing an ON or OFF value. Suppose for instance that the system contains a hardwired rule triggered by the presence in the CM of a loving situation. Then the system must contain a certain LM with the following property: *an ON value at any position (x,y) in the LM indicates that register (x,y) of the CM currently denotes a loving situation*. More precisely, such a CM register is white highlighted and has a black-

highlighted neighbor containing the LOVE symbol (recall the Figure). To cut a long story short, the (x,y) register in the LM receives a connection path from the CM's (x,y) register and each neighboring CM register. As a result, ONness at (x,y) in the LM can be viewed as indicating the temporary association of the CM (x,y) register and its black-highlighted neighbor. Therefore, the LM (x,y) register, which is implemented as a connectionist subnetwork, is acting as a binder.

But it is essential to realize that Conposit's RPE is independent of such binders, in the sense that they serve *only* the subconfiguration-*detection* needs of hardwired rules: they are not brought in by the sheer requirement of *representation*. This is underscored by two observations. (1) Rules can analyze data structures by traversing them, through the use of highlighting movements in the CM, without having first detected them by means of the binders in LMs. (2) Later versions of Conposit will contain secondary CMs used for short-term storage purposes only, and having no attached LMs.

On the other hand, Conposit's PSA can be construed as being based on binders — of a highly atypical nature — in the way described in the first section.

SIMULATIONS OF CONPOSIT

The simulations (performed on NASA's Massively Parallel Processor) incorporate timing delays based on crude but reasonable neural implementation assumptions [Barnden 1986]. E.g.: 2 milliseconds is allowed for each logical gating operation in the neural "local circuits" hypothesized to exist in CM registers [Barnden 1986]; speeds of 1 and 10 meters/second are assumed for short and long neural signal travel respectively; and hardwired rule circuitry is generously assumed to be 5cm away from the CM. Therefore, there is a significant transmission delay (5 milliseconds) over long distances.

Barnden (1988a) reports experiments with two versions of Conposit. The first incorporates production rules for commonsense reasoning, one of which can be paraphrased as: *if a person X loves a person Y who loves a person Z (different from X), then X is jealous of Z*. Such rules exercise Conposit's handling of variable bindings. The rule executes in 515 milliseconds of simulation time. The rule deals with any situation satisfying the rule condition (and set up as the initial state of the CM), without massive duplication of circuitry for different bindings of the variables.

The second Conposit version in Barnden (1988a) converts a simplified form of parse tree for an active sentence into the parse tree for the corresponding passive sentence, in 1063 milliseconds of simulation time. The conversion parallels a task performed by the connectionist production system of Touretzky (1986). The initial tree is given as the initial state of the CM, and Conposit replaces it by the new tree. The rule has no massive duplication of circuitry for different choices of word in the sentence.

The Conposit version described in Barnden (1988b) engages in syllogistic reasoning, by embodying some core aspects of the Johnson-Laird's "mental model" theory [see e.g. Johnson-Laird & Bara 1984]. (Conposit could also accommodate more conventional logical processing of syllogisms.) An example syllogism is: *"Some athletes are beekeepers; all beekeepers are chemists; therefore, some athletes are chemists"*. This is represented in the CM by propositional subconfigurations analogous to those in the Figure above. Conposit's syllogism-processing rules (which involve no massive duplication of circuitry for different choices of object categories in syllogisms) use the syllogism premises to construct a random Johnson-Laird mental model — a sort of example situation involving several athelete, beekeeper and chemist "tokens". A rule then checks whether the syllogism conlcusion is consistent with the mental model. Conposit can run through a syllogism (which involves 6 rule firings, and the creation of 15 tokens on average) in about 2.5 seconds of simulation time. This appears to be fast enough for psychological plausibility, judging by the times allowed to human subjects in Johnson-Laird's experiments.

CONCLUSION

In response to the "challenge" mentioned in the first section, McDermott (1986) has advocated systems that contain symbols — reproducible patterns of network-node activity that can have distinct simultaneous occurrences. Conposit is based on just such symbols. Their use in a Relative-Position Encoding technique, combined with Pattern-Similarity Association, leads to a (quasi-)connectionist system capable of complex high-level symbolic information processing.

The temporary-association methods used by Conposit are particular points in a rich space of techniques for the analysis of which a general unifying framework would be beneficial. The following comments explain the intuitive basis of a general framework that I am currently devising, to accommodate the classes of technique discussed in the first section, and thereby to illuminate their relationships. In particular, the relationship of binder techniques and PSA techniques needs further clarification.

The various techniques discussed embody ways of *TEMPORARILY exploiting PERMANENT structure to achieve TEMPORARY association*. In the weight-change case the permanent structure is some set of connection chains in the total network, and the association is achieved by temporarily exploiting connection chains in the sense of putting them into particular temporary states. The binder case is similar, but the exploitation consists of changing the activation of intermediate nodes in connection chains. In the pattern-relationship case, at least in the special case of PSA, the permanent structure exploited is the appropriate type(s) of similarity among activation patterns, and the exploitation is the placing of similar patterns in suitable network regions. In the positional case, the permanent structure exploited is the way some register-like subnetworks are arranged in the total system architecture, and the exploitation consists of the placing of suitable activation patterns in the registers.

It is illuminating to apply this unifying view of temporary association (as temporary exploitation of permanent structure) to the way it is achieved in primary storage in ordinary computers. The associations are achieved in temporary states of the store that exploit various types of permanent structure. The *sequential allocation* technique for data structuring, where for instance the items in an array are placed in sequence in some set of consecutive locations, exploits the total order on the set of storage locations. The *pointer* or *linking* technique for achieving data structuring exploits the (partial) function from bit-strings, construed as addresses, to locations. A simple form of *content addressing*, in which two locations containing the same (sub)string of bits may be considered to be currently associated, exploits similarity relationships among bit-strings. In sum, sequential allocation, content addressing and linking temporarily exploit permanent structure defined over the set of locations, the set of possible bit-strings, and the mapping from bit-strings to locations, respectively.

It should be obvious how this view of computers can be modified to cover RPE and PSA in Conposit, taking the CM to play the role of the store and ignoring the linking technique. I am hoping to extend the view to cover positional and pattern-relationship techniques more generally, and to encompass also the binder and weight-change classes of technique. Particular tasks to be faced in constructing such a general framework will be to relate it to certain other general frameworks, such as Smolensky's (1987) account of roles, fillers, and variables, and Walters' (1987) account of connectionist information encoding.

ACKNOWLEDGMENT

I am indebted to NASA for unlimited free access to the Massively Parallel Processor at the Goddard Space Flight Center, through my membership in the MPP Working Group.

REFERENCES

Barnden, J.A. On association techniques in neural representation schemes. *Procs. 5th Conf. of the Cognitive Science Society*, Rochester, NY, 1983.

Barnden, J.A. On short-term information processing in connectionist theories. *Cognition and Brain Theory, 7* (1), 1984.

Barnden, J.A. Diagrammatic short-term information processing by neural mechanisms. *Cognition and Brain Theory, 7* (3&4), 1985.

Barnden, J.A. Complex cognitive information-processing: a computational architecture with a connectionist implementation. Tech. Rep. 211, Computer Science Dept., Indiana University, Bloomington, IN. December 1986.

Barnden, J.A. Simulation of an array-based neural net model. In *Proceedings of the First Symposium on the Frontiers of Massively Parallel Scientific Computation*. NASA Conference Publication 2478. 1987.

Barnden, J.A. High-level reasoning in a quasi-connectionist register-array model. Submitted to European Conference on Artificial Intelligence, 1988a.

Barnden, J.A. Commonsense syllogistic reasoning in CONPOSIT, a quasi-connectionist register-array model. Submitted to National Conference on Artificial Intelligence, 1988b.

Charniak, E. & Santos, E. A connectionist context-free parser which is not context-free, but then it is not really connectionist either. *Procs. 9th Annual Conf. of the Cognitive Science Society,* 1987.

Cottrell, G.W. Connectionist parsing. *Procs. 7th Annual Conf. of the Cognitive Science Society,* 1985.

Derthick, M. A connectionist architecture for representing and reasoning about structured knowledge. *Procs. 9th Annual Conf. of the Cognitive Science Society,* 1987.

Feldman, J.A. Dynamic connections in neural networks. *Biological Cybernetics, 46,*, 27-39, 1982.

Hinton, G.E. Representing part-whole hierarchies in connectionist networks. Manuscript, Computer Science Department, Carnegie Mellon University, Pittsburg, Penn., 1987.

Johnson-Laird, P.N. & Bara, B.G. Syllogistic inference. *Cognition, 16* (1), pp.1-61, 1984.

McClelland, J.L. The programmable blackboard model of reading. In J.L. McClelland, D.E. Rumelhart and the PDP Research Group, *Parallel Distributed Processing, Vol. II.* MIT Press, Cambridge, Mass. 1986.

McDermott, D. What AI needs from connectionism. Appendix to J.L. McClelland, J.A. Feldman, G. Bower & D. McDermott, "Connectionist models and cognitive science: goals, directions and implications," a report on an NSF workshop, 1986.

Norman, D.A. Refelections on cognition and parallel distributed processing. In J.L. McClelland, D.E. Rumelhart and the PDP Research Group (Eds.), *Parallel Distributed Processing Vol. 2,* Cambridge, Mass.: MIT Press, 1986.

Pollack, J.P. Cascaded back-propagation on dynamic connectionist networks. *Procs. 9th Annual Conf. of the Cognitive Science Society,* 1987.

Smolensky, P. On variable binding and the representation of symbolic structures in connectionist systems. Tech. Rep. CU-CS-355-87, Dept. of Computer Science, Univ. of Colorado, Boulder, 1987.

Touretzky, D.S. Representing and transforming recursive objects in a neural network, or "Trees *Do* Grow on Boltzmann Machines". *Procs. IEEE Conf. on Systems, Man and Cybernetics,* 1986.

Touretzky, D.S. & Geva, S. A distributed connectionist representation for concept structures. *Procs. 9th Annual Conf. of the Cognitive Science Society,* 1987.

Touretzky, D.S. & Hinton, G.E. Symbols among the neurons: details of a connectionist inference architecture. *Procs. 9th Int. Joint Conf. on Artificial Intelligence,* 1985.

Walters, D. Properties of connectionist variable representations. *Procs. 9th Annual Conf. of the Cognitive Science Society,* 1987.