SIMULATIONS OF CONPOSIT, A SUPRA-CONNECTIONIST ARCHITECTURE FOR COMMONSENSE REASONING

John A. Barnden

Computing Research Laboratory, New Mexico State University
Box 3CRL, Las Cruces, NM 88003.

ABSTRACT

A computational architecture called "Conposit" is outlined. Conposit manipulates very-short-term complex symbolic data structures of types that are useful in high-level cognitive tasks such as commonsense reasoning, planning, and natural language understanding. Conposit can be straightforwardly implemented as a large neural/connectionist network, and therefore provides a way of bridging the gap between high-level cognitive information processing and neural networks. Conposit's data structures are, essentially, temporary configurations of symbol occurrences in a 2D array of registers. Each register is implementable as a neural subnetwork whose activation pattern realizes the symbol occurrence. The data structures are manipulated by condition-action rules that are realizable as further neural subnetworks attached to the array. In simulations, Conposit has performed symbolic processing of types previously found difficult for connectionist/neural networks. This paper concentrates on a version of Conposit, simulated on the Massively Parallel Processor, embodying core aspects of Johnson-Laird's mental model theory of human syllogistic reasoning. This version illustrates Conposit's power and flexibility, which arises from unusual data-structure encoding techniques called "Relative-Position Encoding" and "Pattern-Similarity Association"

Keywords: Cognitive Modeling, Commonsense Reasoning, Connectionism, Neural Network, Knowledge Representation, Syllogism, Mental Model.

INTRODUCTION

The challenge presented to connectionism by high-level cognitive processing — which includes commonsense reasoning, planning, and some aspects of natural language understanding — is gaining increasing recognition. The main technical difficulties are listed in Refs. 1-4, 7, 8, and elsewhere in the connectionist literature, and include the well-known variable-binding problem and the problem of accounting for complex, temporary, novel data structures.

Ref. 6 reports experiments with a version of Conposit that 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.

This exercises Conposit's handling of variable bindings.

The version of Conposit described below engages in a particular type of commonsense reasoning, namely syllogistic reasoning, by embodying some core aspects of the Johnson-Laird's "mental model" theory (Refs. 9-12). The main goal of the work was to verify that the techniques developed for other types of processing in Conposit (Refs. 4-7) were flexible enough to be extended in a natural way to the distinctly different type of processing required by the mental model theory — and in fact no new features have had to be added.

Conposit is currently concerned only with short-term processing: there is no adaptive learning capability at present, and long-term memory consists entirely of the fixed set of condition-action rules (but see the suggestions in Refs. 4,7 for a long-term memory of data structures). It is closer to the "localist" than to the "distributed" end of the spectrum of connectionist systems.

BRIEF SKETCH OF CONPOSIT

Conposit is currently defined as a computational architecture whose components can be straightforwardly implemented in connectionist terms. Details are reported in Ref. 7 (or Ref. 4 for an earlier formulation).

In Conposit, a *"Relative-Position Encoding"* technique is used as the foundation for complex *short-term* data structures. These reside in a 32×32 array of *registers*. This array is called the *configuration matrix* (CM). The values in registers are usually rapidly changing. Each register can be implemented as a small connectionist subnet that holds a dynamically changing activity pattern implementing the register's value, and that is connected to neighboring registers and other components.

A register's value consists of a "symbol" and a vector of binary "highlighting flag" values. A symbol may have a specific representational function, such as denoting a particular person or a particular type of relationship among people. Any symbol can be placed in any register, and all registers

CH2649-2/89/0000/0311$01.00 © 1988 IEEE

have the same set of highlighting flags. *Temporary structure is encoded mainly in the adjacency relationships among values in CM registers.* For instance, if a register contains a symbol denoting the class of all possible situations in which one person loves another, and has a certain highlighting flag in the ON state, then any adjacent register that has another specific highlighting flag ON is deemed to represent, temporarily, a specific loving situation.

See, for example, the representation of the proposition that John loves Mary in the upper portion of Figure 1, which shows an 8 × 8 region of the CM.

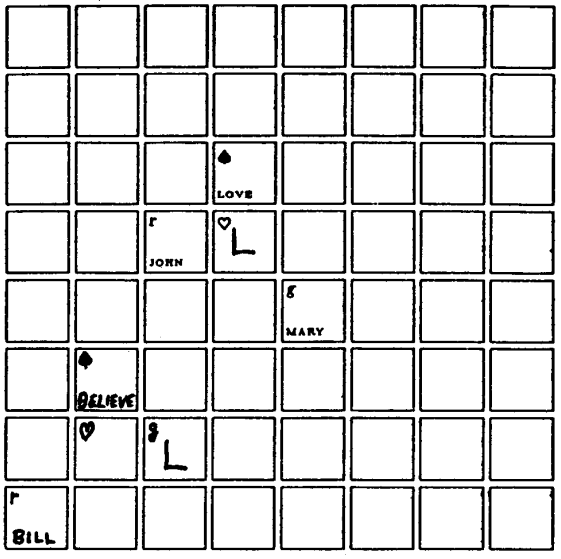

Figure 1. "Bill believes that John loves Mary."

Each square stands for a register, and capitalized words and letters stand for symbols. The word JOHN stands for a symbol denoting a particular person John known to the system. The LOVE symbol denotes the class of all conceivable loving situations. The L symbol may be ignored for now. The registers with no symbol shown contain a null symbol that does not denote anything. The denotations of symbols are considered to be borrowed by the registers they occur in at any moment: a register containing a non-null symbol denotes what that symbol denotes. Hence, in the figure there are registers that — temporarily — denote John, Mary and the love-situations class. The other signs within squares show ON states of highlighting flags, which in this example are all referred to by the names of colors. An 'r' indicates that the register is red-highlighted (i.e. the red flag is currently on); similarly 'g' for green, heart sign for white, and spade sign for black. One important function for highlighting is to help specify the representational relationships temporarily holding between adjacent registers. For instance, a white-highlighted register is deemed to denote a member of the class denoted by any neighboring black-highlighted register. Therefore the upper white register in the figure denotes some love situation. Further, if a register denotes a love situation, then any adjacent red register (here, the one containing JOHN) denotes the "lover" and any adjacent green one (here, the one containing MARY) denotes the "lovee". Note that the *absolute* positions of the symbols and highlighting states are irrelevant, as are the *directions* of the adjacency relationships.

Complex data structures can be split up into pieces by a *shared-symbol association* technique. Shared-symbol association relies on the stipulation that two registers containing the same symbol are considered to represent the same entity. The real power comes from the sharing of variable-like "unassigned symbols". By appearing within a data structure, an unassigned symbol can be viewed as having a temporary denotation dictated by the role of the symbol in the structure. The letter 'L' in Figure 1 indicates an unassigned symbol, which temporarily comes to name the hypothetical loving situation by being in the white-highlighted register in the loving-subconfiguration. The Figure shows how the proposition that *Bill believes that John loves Mary* can be encoded by two separate register-value subconfigurations that are linked by the sharing of the L symbol.

In this shared-symbol association technique, two or more registers contain the *same* symbol, and to that extent contain similar activity patterns at the connectionist level of description. The notion of similarity here is simple and all-or-none (i.e. not graded), but other versions of the technique could be based on more sophisticated, and perhaps graded, notions of similarity of connectionist activity patterns. Shared-symbol association is thus a simple instance of the class of "Pattern-Similarity Association" techniques, which are discussed briefly in Ref. 6.

The processing of the short-term data structures in the CM is performed by internal "circuitry" (i.e. system components that are mapped straightforwardly into a connectionist implementation) mediating mainly neighbor-neighbor interaction within the CM, and external "circuitry" outside the CM but attached to it. 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 isomorphic to the CM (Refs. 4,7), and can in response send complex sequences of signals to the CM. A rule can embody conditionals testing the CM state, loops, and a simple form of non-recursive routine calling. A rule operates on the CM in a highly SIMD, register-local, parallel fashion: each action on the CM is performed by sending to each register an identical "command signal" in parallel, whereupon different registers change state differently, according to their own current states and those of their immediate neighbors.

A command signal can have one of a number of effects, such as making each register that has specified highlighting flags ON or OFF change the states of some flags, and/or accept a new symbol value, and/or broadcast its symbol value

to the other registers (via a central relay station attached to the CM and called the Parallel Distributor). It is also possible for a signal only to have an effect on a single, randomly chosen register with specified highlighting, rather than on each such register. A command signal may also require that, for a register to respond, either some or all of its neighboring registers be in a specified highlighting state. Refs. 4, 7 detail 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 the CM.

A tentative mapping of the model to connectionist networks that appear to be biologically reasonable is sketched in Refs. 4, 7. In particular, it is suggested that the CM could be realized as a localized group of thin cortical columns. It is this suggestion that motivates the choice of dimension two and size 32×32 for the CM (see Ref. 4). A non-biological version of the approach could be based on a CM of other dimensions and sizes.

JOHNSON-LAIRD AND SYLLOGISMS

Consider the syllogism

Some chemists are beekeepers.
All beekeepers are householders.
Therefore, some chemists are householders.

To simplify a little, Johnson-Laird maintains that we make such a syllogistic inference by constructing a mental model of the form illustrated in Figure 2.

C	=	B	=	H
C	=	B	=	H
(C)		(B)	=	(H)
(C)				(H)
				(H)

Figure 2. A Johnson-Laird syllogistic mental model.

This mental model is an abstract data structure made up of "tokens" (shown by the capital letters) and identity links between tokens (shown by the equality signs). There is an arbitrarily selected number of tokens C standing for chemists. An arbitrarily selected proper non-empty subset are related by identity links to beekeeper tokens B, and all beekeeper tokens are so linked to householder tokens H. The parentheses in the figure indicate that the enclosed tokens are optional. The conclusion that some chemists are householders arises from noticing that some chemist tokens are linked by chains of equality tokens to householder tokens. There is much arbitrariness in the construction of a mental model. For instance, the number of tokens in a particular model is arbitrarily chosen, as is the number marked as optional. There is also leeway in how the links are placed. The mental model serves as a sort of internalized, highly abstract "example" situation conforming to the premises of the English syllogism. Naturally, the "conclusion" read off from a mental model might merely be an artifact of the particular example it embodies, and therefore be invalid. In response to this, Johnson-Laird postulates that the system attempts to construct several different mental models conforming to the premises in an attempt to falsify any particular putative conclusion before outputing it. The attempted-falsification process will fail in the present case, but should succeed if in the above syllogism contained "some beekeepers" rather than "all beekepers".

Johnson-Laird's theory is able to explain certain syllogistic preferences, difficulties and errors exhibited by human subjects. He does not specify any implementation of mental models in neural net terms.

JOHNSON-LAIRD SYLLOGISTIC REASONING IN CONPOSIT

Conposit straightforwardly represents mental models, and constructs them from propositional CM subconfigurations that encode syllogism premises. I have not yet addressed the following aspects of Johnson-Laird's approach: (i) the understanding or generation of natural language; (ii) a thorough attempted-falsification process — the current Conposit is given the conclusion, and merely checks its validity with a single model randomly generated from the premises; or (iii) negative premises and conclusions ("no X are Y" and "some X are not Y"), which require special representational and processing features. The correction of the last two deficiencies is not difficult, however, and will be described elsewhere.

Figure 3 shows the CM version of a syllogistic model derived from the premises in the Section 3 syllogism. The CHMS, BKRS and HHS symbols denote the classes of all conceivable chemists, beekeepers and householders respectively. The X1 to X7 are distinct unassigned symbols. Each Johnson-Laird "person token" is implemented as a pair of adjacent CM registers, one of which (the black one) temporarily represents a class of person, and the other of which (the white one, containing an Xi symbol) represents a particular though indefinite member of the class. (Recall the use of white/black adjacent highlighting in the love-situation representation in Figure 1). Each Xi symbol is thereby considered to denote a person for the time being. The Figure shows the person tokens positioned in a regimented way, but in the actual simulation they are randomly positioned in the CM, and the white-highlighted register in each pair is a random neighbor of the black register. The function of Johnson-Laird's identity links is taken over by symbol-sharing, which is therefore being used for its *standard* function of making different CM registers represent the same thing. In the figure an 's' indicates special highlighting signifying that the token is optional.

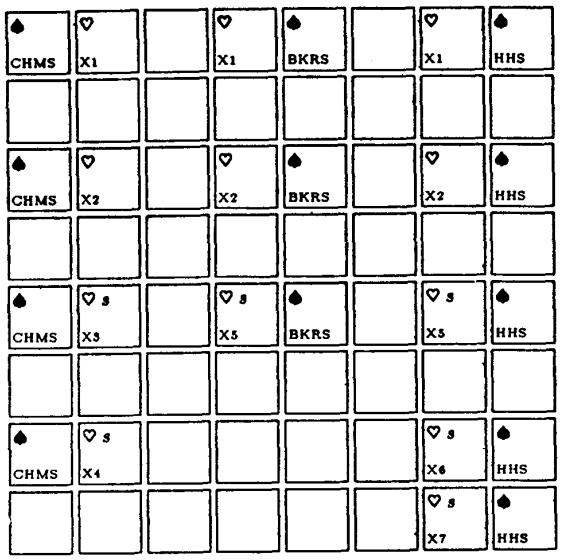

Figure 3. CM version of a syllogistic mental model.

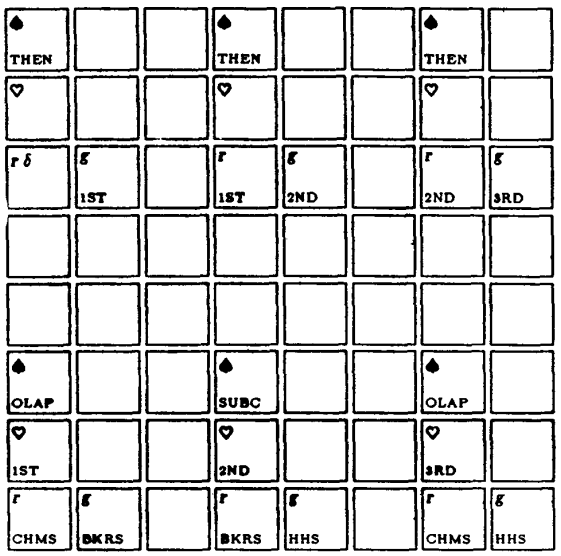

Figure 4. The statement of the syllogism displayed in the text.

The mental model in Figure 3 is constructed from representations, analogous to the one for John loving Mary in Figure 1, of the two premises of the syllogism. These premise representations are shown at the bottom left and bottom middle of Figure 4. Conposit is also given a propositional representation for the syllogism's conclusion (bottom right of Figure 4) and checks that the mental model is consistent with this given conclusion. The OLAP and SUBC symbols denote the classes of all conceivable class-overlap situations and subclass situations respectively. The 1ST, 2ND and 3RD symbols are arbitrary, distinct, unassigned symbols. None of these five symbols is dedicated to syllogistic reasoning. The registers containing 1ST denote the situation of chemists overlapping with beekeepers (i.e. of some chemists being beekeepers). The registers containing 2ND and 3RD are analogously interpreted.

The construction of the mental model has two main phases. A hardwired rule called *Rule_Some* detects the subconfiguration for the first premise (Figure 4, botttom left), and constructs, in a another part of the CM, the chemist and beekeeper tokens in Figure 3. (It creates randomly many chemist tokens, six on average, then constructs beekeeper tokens using the same unassigned symbols as in a random subset of the chemist tokens, and, finally, randomly constructs three extra beekeeper tokens on average.) Another, similar, rule called *Rule_All* detects the subconfiguration for the second premise and constructs some householder tokens with the same unassigned symbols as in the beekeeper tokens, and then constructs some extra householder tokens. Finally, *Rule_Some* comes into play again by detecting the subconfiguration for the conclusion (bottom right of illustration) and checking that there is at least one chemist token and householder token sharing a symbol. In cases where the conclusion is invalid (such as in the amended Section 3 example), Conposit sometimes does and sometimes does not construct a mental model consistent with the conclusion, because of the randomness. It would be trivial to get Conposit to repeat the whole process in an attempt to randomly alight on a falsifying model.

Rule_Some and *Rule_All* work with *any* classes in syllogisms, not just the chemist, beekeeper and householder classes. There is no replication of rule circuitry for the different classes. Achievement of this effect in a more standard type of connectionist system would cause considerable difficulty.

Ref. 4 describes versions of *Rule_Some* and *Rule_All* in complete detail. That paper also describes the rule *Note_Next* that fires three times, once in response to each of the propositional CM subconfigurations at the top of Figure 4. These state the order in which the premise subconfigurations are to be considered. (The THEN symbol denotes the class of all conceivable succession situations.) *Note_Next* moves highlighting of two special sorts around in the CM with the result that *Rule_Some* and *Rule_All* are triggered in the right order.

Rule_Some checks the conclusion in our example as follows. It marks all the white registers in chemist and householder tokens with special highlighting flags "member_of_class1" and "member_of_class2" respectively. Part of this marking process is to spread such highlighting to all registers with the same symbol. All that is left to do is to detect the presence of some register marked with both "member_of_class1" and "member_of_class2". We have here a traditional marker passing process, *but working over highly temporary data structures.*

Simulation Results

Elapsed simulated time depends on values for signal-travel distances, signal-travel speeds, and combinatorial-logic delays (e.g. within CM registers) that are based on broad assumptions about how Conposit could be realized as a biologically reasonable neural net (Ref. 4), rather than just as an abstract connectionist net. The main parameter values are as follows:

distance between rule circuitry and the CM:	50mm
long distance transmission speed:	10mm/ms
basic time for register's response to a signal:	10ms
overhead of random register selection:	5ms

The values of the last two parameters listed appeal to fast non-spike inter-neural communication in local circuits (see Ref. 4 for a discussion). Notice the long distance of 5 centimeters between CM and rule circuitry. The 10mm/ms value appears to be about the maximum speed for transmisison of neural impulses over long distances in cortex.

The following average timings were observed over one set of twelve experiments conducted (one syllogism per experiment).

processing of a whole syllogism	2526ms
detection phase of a rule	98ms
a *Note_Next* execution	76ms
a *Rule_Some/All* execution on first premise	905ms
a *Rule_Some/All* execution on second premise	602ms
a *Rule_Some/All* execution on conclusion	180ms

CONCLUSION

The average syllogism-processing time of about 2.5 seconds seems small enough to be psychologically realistic. It is hard to discern timings for human syllogistic reasoning in Johnson-Laird's experimental reports, partly because of the need for a natural language understanding phase. The experiments all appear to have allowed a time much longer than two and a half seconds. E.g. in the experiments of Ref. 11 subjects were given either ten seconds or as long as they liked. According to figures of Bara (personal communication), the faster human subjects work a simple syllogism in a time comparable to the two and a half seconds needed by Conposit.)

It is probably not biologically plausible for rules like Rule_Some and Rule_All to be *hardwired* as in the current Conposit version, partly because of the difficulty of seeing how the rule circuitry could be developed on the basis of experience. However, the basic processing techniques developed will be central also in more realistic systems in which a high-level production rule such as Rule_Some would itself be a *data structure* in one of a possibly large *set* of CMs (Refs. 4,7). In such systems, which are under investigation, individual rule execution could be *faster* because of faster subconfiguration creation, and because there would be the possibility of massive parallelism among rules in different CMs.

The power and flexibility of Conposit arises from its Relative-Position Encoding and Pattern-Similarity Association techniques for encoding data structures. These techniques are unusual for connectionism, although the are loosely related to methods found elsewhere (see Ref. 6).

ACKNOWLEDGMENTS

This research is being supported in part by USAF under grant AFOSR-88-0215 and by NASA through the MPP Working Group.

REFERENCES

1. Barnden, J.A. On association techniques in neural representation schemes. *Procs. 5th Conf. of the Cognitive Science Society*, Rochester, NY, 1983.
2. Barnden, J.A. On short-term information processing in connectionist theories. *Cognition and Brain Theory, 7* (1), 1984.
3. Barnden, J.A. Diagrammatic short-term information processing by neural mechanisms. *Cognition and Brain Theory, 7* (3&4), 1985.
4. Barnden, J.A. Complex cognitive information-processing: a computational architecture with a connectionist implementation. Tech. Rep. 211, Computer Science Dept., Indiana University. 1986.
5. 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.
6. Barnden, J.A. The right of free association: relative-position encoding for connectionist data structures. *Procs. 10th Annual Conf. of the Cognitive Science Soc.* Hillsdale, N.J.: Lawrence Erlbaum, 1988.
7. Barnden, J.A. The power of some unusual connectionist data-structuring techniques. In J.A. Barnden & J.B. Pollack (Eds), *Advances in Connectionist and Neural Computation Theory, Vol. 1,* Norwood, N.J.: Ablex, to appear.
8. Dyer, M.G. Symbolic NeuroEngineering and natural language processing: a multilevel research approach. In J.A. Barnden & J.B. Pollack (Eds), *Advances in Connectionist and Neural Computation Theory, Vol. 1,* Norwood, N.J.: Ablex, to appear.
9. Johnson-Laird, P.N. *Mental models.* Harvard University Press: Cambridge, Mass., 1983.
10. Johnson-Laird, P.N. Reasoning by rule or model? *Procs. 10th Annual Conf. of the Cognitive Science Soc.* Hillsdale, N.J.: Lawrence Erlbaum, 1988.
11. Johnson-Laird, P.N. & Bara, B.G. Syllogistic inference. *Cognition, 16* (1), pp.1-61, 1984.
12. Johnson-Laird, P.N., Oakhill, J. & Bull, D. Children's syllogistic reasoning. *The Quaterly J. of Experimental Psych., 38A,* 35-58, 1986.