(1)

ONR ANNUAL PRODUCTIVITY REPORT, 01 SEP 1993 31 AUG 1994

PRINCIPAL INVESTIGATOR: Lokendra Shastri

INSTITUTION: International Computer Science Institute

PROJECT TITLE: Spatio-temporal Neural Networks for Vision, Reasoning and Rapid Decision Making (N00014-93-1-1149)

Number of ONR supported:

Papers published in refereed journals: 1
Papers accepted for publication in referred journals: 0
Papers under review for publication in referred journals: 1
Books or book chapters published: 0
Books or book chapters in press: 1

Number of ONR supported patents/inventions filed 0 or granted 0.

Number of presentations:
Invited: 3
Contributed: 1

Trainee Data:
No. of grad. students: 1, Male Non-US Citizen
No. of undergraduates: 1 Male Non-US Citizen

Number, cost and description of equipment items costing more than $1,000 that were purchased on your ONR grant.

None

Awards/Honors to PI and/or to members of PI's research group (please describe):

None

Brief description of all trasitions (or intended transitions) of your ideas or techniques to industry, to military laboratories or to military application.

As mentioned in the attached report, we have tried to leverage our work on a neurally motivated model of reflexive reasoning to design a real-time large-scale knowledge based reasoning system. We have completed a pilot implementation on a CM-5 and experimented with large randomly generated knowledge bases. The pilot system can encode over 300,000 knowledge base items and respond in less than 500 msec. to queries requiring reasoning upto a depth of eight (shallower queries require even less time). I am interested in identifying applications of interest to the military that would benefit from the capabilities of such a real-time large-scale knowledge base system.

Attachment for: ONR ANNUAL PRODUCTIVITY REPORT, 01 SEP 1993 31 AUG 1994

PRINCIPAL INVESTIGATOR: Lokendra Shastri

INSTITUTION: International Computer Science Institute

PROJECT TITLE: Spatio-temporal Neural Networks for Vision, Reasoning and Rapid Decision Making

Progress Summary

A spatio-temporal system for recognizing handprint digit strings was designed and trained to recognize handprinted ZIP codes. The results of our work on a biologically motivated model of reflexive reasoning were used to implement a pilot system for performing rapid reasoning using very large knowledge bases. The pilot system which runs on a 32 node CM-5, can encode over 300,000 items and respond in less than 500 msec. to queries requiring reasoning upto a depth of eight.

Progress Report

We have continued our investigation of the representational capabilities of spatio-temporal networks and their application to reflexive reasoning and pattern recognition. These network use recurrent connections and variable delay links. In addition to the firing rate, the firing time of cells relative to other cells, carries representational significance in these models (the synchronous firing of cells being an important special case).

We finished the design of a spatio-temporal model for handprint digit string recognition. The model was trained to recognize handprinted ZIP codes. In addition to the obvious practical significance, the work furthers our understanding of spatiotemporal models for pattern recognition and demonstrates that the approach offers a natural solution to the problem of shift-invariance, enables a pattern recognition system to handle arbitrarily long inputs and partially solves the segmentation/recognition dilemma. In earlier work we had developed a system for isolated digit recognition and done some preliminary work on extending the system to connected pairs of digits. The additional work extended the system to do full word (ZIP code) recognition. The results of this work are described in an article submitted to the journal *Connection Science* for publication and was the subject of Thomas Fontaine's PhD dissertation (December 1993).

We are also leveraging the results of our reflexive reasoning system based on temporal synchrony to build a system for performing rapid reasoning using very large knowledge bases. The aim is to build a system whose response time is fast enough to support inferencing for a real-time speech understanding system. This means being able to respond to retrieval as well as inferential queries within a few hundred milliseconds. We have a pilot implementation on a 32 node CM-5 that can encode over 300,000 rules, facts, and types and respond to queries whose response requires inferences that are 10 deep in about 500msec. The effectiveness of the implementation can be directly attributed to the constraints on representation and inference suggested by the use of temporal synchrony for expressing dynamic bindings. The result of the CM-5 implementation are described in the enclosed technical report (ICSI TR-94-031). This research is the topic of D.R. Mani's PhD dissertation.

Results of work on our model for reflexive reasoning using temporal synchrony have appeared in *Behavioral and Brain Sciences* and *Connection Science* journals and in the *1993 International Joint Conference on Artificial Intelligence.*

I have begun investigating a possible solution to the "catastrophic interference problem". In brief, the problem is this: If a network that has already been trained to solve task A is trained to solve task B, it forgets the solution to task A unless it is simultaneously retrained on task A. This problem is an inherent weakness of most incremental learning algorithms and is perhaps the biggest impediment in the development of scalable learning systems. The solution being investigated is as follows: *Initially the system focuses on* a small number of categories. After it learns these categories, it tries to identify which features formed in the "hidden layer" play a crucial role in the recognition of these categories. The system freezes these crucial features and as a result they cannot be obliterated during subsequent learning (although they may undergo some fine tuning). These frozen features are however, available to other structures that are learned subsequently to recognize other categories. An important claim is that the set of features will gradually stabilize and learning new categories will get progressively easier and involve combining existing features in the appropriate manner. These ideas are being investigated in the context of training using spatio-temporal to recognize digit strings.

BEHAVIORAL AND BRAIN SCIENCES (1993) 16, 417–494
Printed in the United States of America

Related work

From simple associations to systematic reasoning: A connectionist representation of rules, variables and dynamic bindings using temporal synchrony

Lokendra Shastri
Computer and Information Science Department, University of Pennsylvania, Philadelphia, PA 19104
Electronic mail: *shastri@central.cis.upenn.edu*

Venkat Ajjanagadde
Wilhelm-Schickard-Institut, University of Tuebingen, Sand 13 W-7400 Tuebingen, Germany
Electronic mail: *nnsaj01@mailserv.zdv.uni-tuebingen.de*

Abstract: Human agents draw a variety of inferences effortlessly, spontaneously, and with remarkable efficiency – as though these inferences were a reflexive response of their cognitive apparatus. Furthermore, these inferences are drawn with reference to a large body of background knowledge. This remarkable human ability seems paradoxical given the complexity of reasoning reported by researchers in artificial intelligence. It also poses a challenge for cognitive science and computational neuroscience: How can a system of simple and slow neuronlike elements represent a large body of systemic knowledge and perform a range of inferences with such speed? We describe a computational model that takes a step toward addressing the cognitive science challenge and resolving the artificial intelligence paradox. We show how a connectionist network can encode millions of facts and rules involving *n*-ary predicates and variables and perform a class of inferences in a few hundred milliseconds. Efficient reasoning requires the rapid representation and propagation of dynamic bindings. Our model (which we refer to as SHRUTI) achieves this by representing (1) dynamic bindings as the synchronous firing of appropriate nodes, (2) rules as interconnection patterns that direct the propagation of rhythmic activity, and (3) long-term facts as temporal pattern-matching subnetworks. The model is consistent with recent neurophysiological evidence that synchronous activity occurs in the brain and may play a representational role in neural information processing. The model also makes specific psychologically significant predictions about the nature of reflexive reasoning. It identifies constraints on the form of rules that may participate in such reasoning and relates the capacity of the working memory underlying reflexive reasoning to biological parameters such as the lowest frequency at which nodes can sustain synchronous oscillations and the coarseness of synchronization.

Keywords: binding problem; connectionism; knowledge representation; long-term memory; neural oscillations; reasoning; short-term memory; systematicity; temporal synchrony; working memory

1. Introduction

The ability to represent and reason with a large body of knowledge in an effective and systematic manner is a central characteristic of cognition. This is borne out by research on artificial intelligence and cognitive science, which suggests that reasoning underlies even the most commonplace intelligent behavior. For example, language understanding, a task we usually perform rapidly and effortlessly, depends upon our ability to make predictions, generate explanations, and recognize speakers' plans.[1] To appreciate the richness and speed of human reasoning, consider the following example derived from Schubert (1989). Imagine a person reading a variation of the Little Red Riding Hood (LRRH) story, in which the wolf intends to eat LRRH in the woods. The reader is at the point in the story where the wolf, who has followed LRRH into the woods, is about to attack her. The next sentence reads: "The wolf heard some woodcutters nearby and so he decided to wait." It seems reasonable to claim that the reader will understand this sentence spontaneously and without conscious effort. However, a careful analysis suggests that even though the reader remains unaware of it, understanding this sentence requires a chain of reasoning that may be described informally as follows (parenthetical text identifies the background knowledge that might mediate the reasoning process):

The wolf will approach LRRH (to eat something you have to be near it); LRRH will scream (because a child is scared by an approaching wild animal); upon hearing the scream the woodcutters will know that a child is in danger (because a child's screaming suggests that she is in danger); the woodcutters will go to the child (people want to protect children in danger, and in part this involves determining the source of the danger); the woodcutters will try to prevent the wolf from attacking LRRH (people want to protect children); in doing so the woodcutters may hurt the wolf (preventing an animal from attacking may involve physical force); so the wolf decides to wait (because an animal does not want to get hurt).

One could argue that some of the steps in this reasoning process are precompiled or "chunked," but it would be unreasonable to claim that the entire chain of reasoning can be construed as direct retrieval or even a single-step inference. Hence, in addition to accessing lexical items, parsing, and resolving anaphoric reference, some computation similar to the above chain of reasoning must occur when the sentence in question is processed. As another example, consider the sentence "John seems to have suicidal tendencies; he has joined the Colombian drug enforcement agency." In spite of its being novel, we can understand the sentence spontaneously and without conscious effort. This sentence, however, could not have been understood without using background knowledge and dynamically inferring that joining the Colombian drug enforcement agency has dangerous consequences, and since John probably knows this, his decision to join the agency suggests that he has suicidal tendencies.

As the above examples suggest, we can draw a variety of inferences rapidly, spontaneously, and without conscious effort – as though they were a reflexive response of our cognitive apparatus. Let us accordingly describe such reasoning as *reflexive* (Shastri 1990).[2] Reflexive reasoning may be contrasted with *reflective* reasoning, which requires reflection, conscious deliberation, and often an overt consideration of alternatives and weighing of possibilities. Reflective reasoning takes longer and often requires the use of external props such as a paper and pencil. Examples of such reasoning are solving logic puzzles, doing cryptarithmetic, or planning a vacation.[3]

Our remarkable ability to perform reflexive reasoning poses a challenge for cognitive science and neuroscience: How can a system of simple and slow neuronlike elements represent a large body of systematic knowledge and perform a range of inferences with such speed? With nearly 10^{12} computing elements and 10^{15} interconnections, the brain's capacity for encoding, communicating, and processing information seems overwhelming. But if the brain is extremely powerful, it is also extremely limited: First, neurons are slow computing devices. Second, they communicate relatively simple messages that can encode only a few bits of information. Hence a neuron's output cannot encode names, pointers, or complex structures.[4] Finally, the computation performed by a neuron is best described as an analog spatio-temporal integration of its inputs. The relative simplicity of a neuron's processing ability with reference to the needs of symbolic computation, and the restriction on the complexity of messages exchanged by neurons, impose strong constraints on the nature of neural representations and processes (Feldman 1989; Feldman & Ballard 1982; Shastri 1991). [See also Feldman: "Four frames suffice: A provisional model of vision and space" *BBS* 8(2) 1985; Ballard: "Cortical connections and parallel processing: Structure and function" *BBS* 9(1) 1986.] As we discuss in section 2, a reasoning system must be capable of encoding systematic and abstract knowledge and instantiating it in specific situations to draw appropriate inferences. This means that the system must solve a complex version of the variable-binding problem (see Section 2 and Feldman 1982; von der Malsburg 1986). In particular, the system must be capable of representing composite structures in a dynamic fashion and systematically propagating them to instantiate other composite structures. This turns out to be a difficult problem for neurally motivated models. As McCarthy (1988) observed, most connectionist systems suffer from the "unary or even propositional fixation" with their representational power restricted to unary predicates applied to a fixed object. Fodor and Pylyshyn (1988a) have even questioned the ability of connectionist networks to embody systematicity and compositionality.

1.1. Reflexive reasoning: Some assumptions, observations and hypotheses

Reflexive reasoning occurs with reference to a large body of long-term knowledge. This knowledge forms an integral part of an agent's conceptual representation and is retained for a considerable period of time once it is acquired. We wish to distinguish long-term knowledge from short-term as well as medium-term knowledge. By the last we mean knowledge that persists longer than short-term knowledge and may be remembered for days or even weeks. Such medium-term knowledge, however, may be forgotten without being integrated into the agent's long-term conceptual representation. The distinction between medium- and long-term knowledge is not arbitrary and seems to have a neurological basis. It has been suggested that medium-term memories are encoded via long-term potentiation (LTP) (Lynch 1986), and some of them subsequently converted into long-term memories and encoded via essentially permanent structural changes (see, e.g., Marr 1971; Squire 1987; Squire & Zola-Morgan 1991).

An agent's long-term knowledge base (LTKB) encodes several kinds of knowledge. These include specific knowledge about particular entities, relations, events, and situations, and general systematic knowledge about the regularities and dependencies in the agent's environment. For example, an agent's LTKB may contain specific knowledge such as "Paris is the capital of France" and "Susan bought a Rolls-Royce," as well as systematic and instantiation-independent knowledge such as "if one buys something then one owns it." We will refer to specific knowledge as *facts*, and general instantiation-independent knowledge as *rules* (note that by a *rule* we do not mean a "rule of inference" such as *modus ponens*). The LTKB may also include knowledge about the attributes of features of concepts and the superordinate/subordinate relations among concepts, and also procedural knowledge such as "how to mow a lawn."

In discussing the LTKB we are focusing on representational adequacy, that is, the need to represent entities, relations, inferential dependencies, and specific as well as general knowledge. The expressiveness implied by this generic specification, however, is sufficient to represent knowledge structures such as *frames* (Minsky 1975), *scripts* (Schank & Abelson 1977), and *productions* or *if-then* rules (Newell & Simon 1972).

A serious attempt at compiling commonsense knowledge suggests that the LTKB may contain as many as 10^8 items (Guha & Lenat 1990). This should not be very surprising given that it must include, besides other things, our knowledge of naive physics and naive psychology; facts about ourselves, our family, and friends; facts about history and geography; our knowledge of artifacts; sports, art, and music trivia; and our models of social and civic interactions.

1.1.1. Space and time constraints on a reflexive reasoner. Given that there are about 10^{12} cells in the brain, the expected size of the LTKB (10^8) rules out any encoding scheme whose node requirement is quadratic (or higher) in the size of the LTKB.[5] In view of this we adopt the working hypothesis that *the node requirement of a model of reflexive reasoning should be no more than linear in (i.e., proportional to) the size of the LTKB*. This is a reasonable hypothesis. Observe that (1) a node in an idealized computational model may easily correspond to a hundred or so actual cells, and (2) the number of cells available for encoding the LTKB can only be a fraction of the total number of cells.

We believe that although the size of an agent's LTKB increases considerably from, say, age 10 to 30, the time taken by an agent to understand natural language does not. This leads us to suspect that the time taken by an episode of reflexive reasoning does not depend on the overall size of the LTKB but only on the complexity of the particular episode of reasoning. Hence we adopt the working hypothesis that *the time required to perform reflexive reasoning is independent of the size of the LTKB*.[6]

The independence of (1) the time taken by reflexive reasoning and (2) the size of the LTKB implies that reflexive reasoning is a parallel process and involves the simultaneous exploration of a number of inferential paths. Hence, a model of reflexive reasoning must be parallel at the level of rule application and reasoning, that is, it must support knowledge-level parallelism. This is a critical constraint and one that is not necessarily satisfied by a connectionist model simply because it is "connectionist" (see also Sumida & Dyer 1989).

We understand written language at the rate of somewhere between 150 and 400 words per minute (Carpenter & Just 1977). In other words, we can understand a typical sentence in a matter of one to two seconds. Given that reflexive reasoning occurs during language understanding, it follows that *episodes of reflexive reasoning may take as little as a few hundred milliseconds.*

1.1.2. Reflexive reasoning is limited reasoning. Complexity theory rules out the existence of a general-purpose reasoning system that derives all inferences efficiently. This entails that there must exist constraints on the class of reasoning that may be performed in a reflexive manner. Not surprisingly, cognitive agents can perform only a limited class of inferences with extreme efficiency. Naturally, we expect that the representational and reasoning ability of the proposed system will also be constrained and limited in a number of ways. However, we would like the strengths and limitations of the system to be psychologically plausible and to mirror some of the strengths and limitations of human reasoning.

1.2. Computational constraints

Connectionist models (Feldman & Ballard 1982; Rumelhart & McClelland 1986) are intended to emulate the information-processing characteristics of the brain – albeit at an abstract computational level – and to reflect its strengths and weaknesses. Typically, a node in a connectionist network corresponds to an idealized neuron, and a link corresponds to an idealized synaptic connection. Let us enumerate some core computational features of connectionist models: (1) Nodes compute very simple functions of their inputs. (2) They can only hold limited state information – while a node may maintain a scalar "potential," it cannot store and selectively manipulate bit strings. (3) Node outputs do not have sufficient resolution to encode symbolic names or pointers. (4) There is no central controller that instructs individual nodes to perform specific operations at each step of processing.

1.3. A preview

We discuss the variable-binding problem as it arises in the context of reasoning and describe a neurally plausible solution to this problem. The solution involves maintaining and propagating dynamic bindings using synchronous firing of appropriate nodes. We show how our solution leads to a connectionist knowledge representation and reasoning system (which we call SHRUTI, see Response, Note 1) that can encode a large LTKB consisting of *facts* and *rules* involving *n-ary predicates* and *variables*, and perform a broad class of reasoning with extreme efficiency. Once a query is posed to the system by initializing the activity of appropriate nodes, the system computes an answer automatically and in time proportional to the length of the shortest chain of reasoning leading to the conclusion. The ability to reason rapidly is a consequence, in part, of the system's ability to maintain and propagate a large number of dynamic bindings simultaneously.

The view of information processing implied by the proposed system is one where (1) reasoning is the transient but systematic propagation of a *rhythmic* pattern of activity, (2) each entity in the dynamic memory is a phase in this rhythmic activity, (3) dynamic bindings are represented as the *synchronous* firing of appropriate nodes, (4) long-term facts are subnetworks that act as temporal pattern matchers, and (5) rules are interconnection patterns that cause the propagation and transformation of rhythmic patterns of activity.

We cite neurophysiological data that suggest that the basic mechanisms proposed for representing and propagating dynamic variable bindings, namely, the propagation of rhythmic patterns of activity and the synchronous activation of nodes, exist in the brain and appear to play a role in the representation and processing of information.

Our system predicts a number of constraints on reflexive reasoning that have psychological implications. These predictions concern the capacity of the working memory underlying reflexive reasoning (WMRR) and the form of rules that can participate in such reasoning. The predictions also relate the capacity of the WMRR and the time it would take to perform one step of reasoning to biological parameters such as the lowest frequency at which nodes can sustain synchronous oscillations, the coarseness of synchronization, and the time it takes connected nodes to synchronize. By choosing biologically plausible system parameters, we show that it is possible for a system of neuronlike elements to encode millions of facts and rules and yet perform multistep inferences in a few hundred milliseconds.

Reasoning is the spontaneous and natural outcome of the system's behavior. The system does not apply syntactic rules of inference such as *modus ponens*. There is no separate interpreter or inference mechanism that manipulates and rewrites symbols. The network encoding of the LTKB is best viewed as a vivid internal model of the agent's environment, where the interconnections between (internal) representations directly encode the dependencies between the associated (external) entities. When the nodes in this model are activated to reflect a given state of affairs in the environment, the model spontaneously simulates the behavior of the external world and in doing so makes predictions and draws inferences.

The representational and inferential machinery developed in this work has wider significance and can be applied to other problems whose formulation requires the expressive power of *n*-ary predicates, and whose solution requires the rapid and systematic interaction between long-term and dynamic structures. Some examples of such problems are (1) parsing and the dynamic linking of syntactic and semantic structures during language processing, and (2) model-based visual object recognition requiring the dynamic representation and analysis of spatial relations between objects and/or parts of objects. Recently, Henderson (1992) has proposed the design of a natural language parser based on our computational model.

1.4. Caveats

Our primary concern has been to extend the representational and inferential power of neurally plausible (connectionist) models and to demonstrate their scalability. We are also concerned that the strengths and limitations of our system be psychologically plausible. However, our aim has not been to model data from specific psychological experiments. What we describe is a partial model of reflexive reasoning. It demonstrates how a range of reasoning can be performed in a reflexive manner, and it also identifies certain types of reasoning that cannot be performed in a reflexive manner. Our system, however, does not model all aspects of reflexive reasoning. For example, we focus primarily on declarative and semantic knowledge and do not model reflexive analogical reasoning, or reflexive reasoning involving episodic memory (Tulving 1983) and imagery. We do not say much about what the actual contents of an agent's LTKB ought to be, nor do we provide a detailed answer to the question of learning. We do, however, discuss in brief how specific facts may be learned and existing rules modified (sect. 10.6). Neural plausibility is an important aspect of this work – we show that the proposed system can be realized by using neurally plausible nodes and mechanisms, and we investigate the consequences of choosing biologically motivated values of system parameters. Needless to say, what we describe is an idealized computational model and it is not intended to be a blueprint of how the brain encodes an LTKB and performs reflexive reasoning.

1.4.1. An outline of the paper. Section 2 discusses the dynamic-binding problem in the context of reasoning. Section 3 presents our solution to this problem and the encoding of long-term rules and facts. Section 4 describes a reasoning system capable of encoding an LTKB and answering queries on the basis of the encoded knowledge. The interface of the basic reasoning system with an *IS-A* hierarchy that represents entities, types (categories), and the super-/subordinate concept relations between them is described in section 5. Section 6 discusses a solution to the multiple instantiation problem. Section 7 discusses the biological plausibility of our system and identifies neurally plausible values of certain system parameters. Section 8 points out the psychological implications of the constraints on reflexive reasoning suggested by the system. Section 9 discusses related connectionist models and the marker-passing system NETL. Finally, section 10 discusses some open problems related to integrating the proposed reflexive-reasoning system with an extended cognitive system. Certain portions of the text are set in small type. These cover detailed technical material and may be skipped without loss of continuity.

2. Reasoning and the dynamic-binding problem

Assume that an agent's LTKB embodies the following rules:[7]

1. If someone gives a recipient an object then the recipient comes to own that object.
2. Owners can sell what they own.

Given the above knowledge, an agent would be capable of inferring "Mary owns Book1" and "Mary can sell Book1" on being told "John gave Mary Book1." A connectionist reasoning system that embodies the same knowledge should also be capable of making similar inferences and, hence, exhibiting the following behavior: If the network's pattern of activity is initialized to represent the fact "John gave Mary Book1," then very soon its activity should evolve to include the representations of the "Mary owns Book1" and "Mary can sell Book1."

Let us point out that the knowledge embodied in a rule may be viewed as having two distinct aspects. A rule specifies a systematic correspondence between the arguments of certain "predicates" (where a predicate may be thought of as a relation, a frame, or a schema). For example, rule (1) specifies that a "give" event leads to an "own" event where the *recipient* of "give" corresponds to the *owner* of "own," and the *object* of "give" corresponds to the *object* of "own." Let us refer to this aspect of a rule as *systematicity*.[8] The second aspect of the knowledge embodied in a rule concerns the *appropriateness* of the specified argument correspondence in a given situation, depending upon the types (or features) of the argument fillers involved in that situation. Thus appropriateness may capture type restrictions that argument fillers must satisfy in order for a rule to fire. It may also indicate type preferences and provide a graded measure of a rule's applicability in a given situation on the basis of the types of the argument fillers in that situation.

We will first focus on the problems that must be solved in order to incorporate systematicity in a connectionist system. In section 5 we will discuss how the solutions proposed to deal with systematicity may be augmented to incorporate appropriateness and represent context-dependent rules that are sensitive to the types of the argument fillers.

If we focus on systematicity, then rules can be succinctly described by using the notation of first-order logic. For example, rules (1) and (2) can be expressed as the following first-order rules:

$$\forall x,y,z\ [give(x,y,z) \Rightarrow own(y,z)] \qquad (1)$$

$$\forall u,v\ [own(u,v) \Rightarrow can\text{-}sell(u,v)] \qquad (2)$$

where *give* is a three-place predicate with arguments: *giver*, *recipient*, and *give-object*; *own* is a two-place predicate with arguments: *owner* and *own-object*; and *can-sell* is also a two-place predicate with arguments: *potential-seller* and *can-sell-object*. The use of quantifiers and variables allows the expression of general, instantiation-independent knowledge and helps in specifying the systematic correspondence between predicate arguments.[9] A fact may be expressed as a predicate instance (atomic formula). For example, the fact "John gave Mary Book1" may be expressed as *give(John, Mary, Book1)*.

A connectionist network must solve three technical problems in order to incorporate systematicity. We discuss these problems in the following three sections.

2.1. Dynamic representation of facts: Instantiating predicates

A reflexive-reasoning system should be capable of representing facts in a rapid and dynamic fashion. Observe that the reasoning process generates inferred facts dynamically and the reasoning system should be capable of representing these inferred facts. Furthermore, the reasoning system must interact with other processes that communicate facts and pose queries to it, and the system should be capable of dynamically representing such facts and queries.

The dynamic representation of facts poses a problem for standard connectionist models. Consider the fact *give(John, Mary, Book1)*. This fact cannot be represented dynamically by simply activating the representations of the arguments *giver*, *recipient*, and *give-object*, and the constituents "John," "Mary," and "Book1." Such a representation would suffer from cross-talk and would be indistinguishable from the representations of *give(Mary, John, Book1)* and *give(Book1, Mary, John)*. The problem is that this fact – like any other instantiation of an *n*-ary predicate – is a composite structure: it does not merely express an association between the constituents "John," "Mary," and "Book1," rather it expresses a specific relation wherein each constituent plays a distinct role. Thus a fact is essentially a collection of *bindings* between predicate arguments and fillers. For example, the fact *give(John, Mary, Book1)* is the collection of argument-filler bindings (*giver* = *John*, *recipient* = *Mary*, *give-object* = *Book1*). Hence representing a dynamic fact amounts to representing, dynamically, the appropriate bindings between predicate arguments and fillers.

The dynamic representation of facts should also support the simultaneous representation of multiple facts such as *give(John, Mary, Book1)* and *give(Mary, John, Car3)* without "creating" ghost facts such as *give(Mary, John, Book1)*.

2.1.1. Static versus dynamic bindings.

A connectionist encoding that represents the bindings associated with the fact *give(John, Mary, Book1)* without cross-talk is illustrated in Figure 1 (cf. Shastri 1988b; Shastri & Feldman 1986). Each triangular *binder* node binds the appropriate filler to the appropriate argument and the *focal* node *give-23* provides the requisite grouping between the set of bindings that make up the fact. The binder nodes become active on receiving two inputs and thus serve to retrieve the correct filler, given a fact and an argument (and vice versa). Such a static encoding, using physically interconnected nodes and links to represent argument-filler bindings, is suitable for representing stable and long-term knowledge, because the required focal and binder nodes may be learned (or recruited) over time in order to represent new but stable bindings of constituents.[10] This scheme, however, is implausible for representing bindings required to encode dynamic structures that will arise during language understanding and visual processing. Such dynamic bindings may have to be represented very rapidly – within a hundred milliseconds – and it is unlikely that there exist mechanisms that can support widespread structural changes and growth of new links within such time scales. An alternative would be to assume that interconnections between *all possible* pairs of arguments and fillers already exist. These links normally remain "inactive" but the appropriate subset of these links becomes "active" temporarily to represent dynamic bindings (Feldman 1982; von der Malsburg 1986). This approach, however, is also problematic because the number of all possible argument-filler bindings is extremely large, and having preexisting structures for representing all

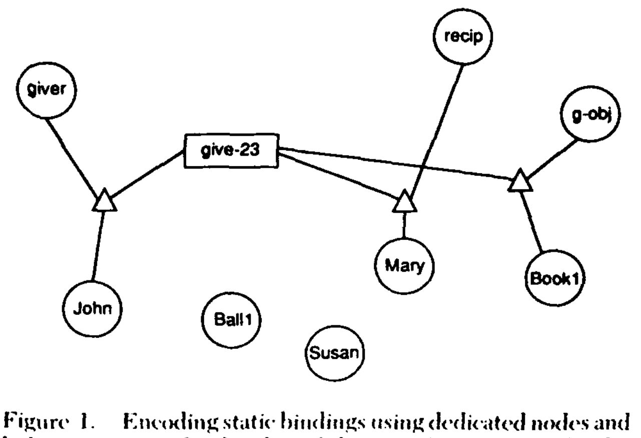


Figure 1. Encoding static bindings using dedicated nodes and links: *give-23* is a *focal* node and the triangles represent *binder* nodes.

these bindings will require a prohibitively large number of nodes and links. Techniques for representing argument-filler bindings on the basis of the von Neumann architecture also pose difficulties because they require communicating names or pointers of fillers to appropriate argument nodes and vice versa. As pointed out earlier, the storage and processing capacity of nodes as well as the resolution of their outputs is not sufficient to store, process, and communicate names or pointers.

2.2. Inference, propagation of dynamic bindings and the encoding of rules

The second technical problem that a connectionist reasoning system must solve concerns the dynamic generation of inferred facts. For example, starting with a dynamic representation of *give(John, Mary, Book1)*, the state of network encoding rules (1) and (2) should evolve rapidly to include the dynamic representations of the inferred facts: *own(Mary, Book1)* and *can-sell(Mary, Book1)*. This process should also be free of cross-talk and not lead to spurious bindings.

Generating inferred facts involves the systematic propagation of dynamic bindings in accordance with the rules embodied in the system. A rule specifies antecedent and consequent predicates and a correspondence between the arguments of these predicates. For example, the rule $\forall x,y,z$ [*give*$(x,y,z) \Rightarrow$ *own*(y,z)] specifies that a *give* event results in an *own* event wherein the *recipient* of a *give* event corresponds to the *owner* of an *own* event and the *give-object* of a *give* event corresponds to the *own-object* of an *own* event. An application of a rule (i.e., a step of inference) therefore amounts to taking an instance of the antecedent predicate(s) and creating, dynamically, an instance of the consequent predicate, with the argument bindings of the latter being determined by applying the argument correspondence specified in the rule to the argument bindings of the former. Thus the application of the rule $\forall x,y,z$ [*give*$(x,y,z) \Rightarrow$ *own*(y,x)], in conjunction with an instance of *give*, *give(John, Mary, Book1)*, creates an instance of *own* with the bindings (*owner* = *Mary*, *own-object* = *Book1*). These bindings constitute the inferred fact *own(Mary, Book1)*. Once the representation of an inferred fact is established, it may be used in conjunction with other domain rules to create other inferred facts. Such a chain of inference may lead to a proliferation of inferred facts and the associated dynamic bindings.

2.3. Encoding long-term facts

In addition to encoding domain rules such as (1) and (2), a connectionist reasoning system must also be capable of encoding facts in its LTKB and using them during recall, recognition, query answering, and reasoning. For example, we expect our system to be capable of encoding a fact such as "John bought a Rolls-Royce" in its LTKB and using it to answer rapidly the query Did John buy a Rolls-Royce? We also expect it to use this fact in conjunction with other knowledge to answer rapidly queries such as Does John own a car? Observe that storing a long-term fact would require storing the associated bindings as a static long-term structure. This structure should interact with dynamic bindings and recognize those that match it.

2.4. Dynamic binding and categorization

As discussed at the beginning of section 2, the appropriateness of a rule in a specific situation may depend on the types/features of the argument fillers involved in that situation. Thus categorization plays a crucial role in the propagation of dynamic bindings during reasoning. Consider the rule: $\forall x,y$ *walk-into*$(x,y) \Rightarrow$ *hurt*(x) (i.e., If one walks into something then one gets hurt). As stated, the rule only encodes systematicity and underspecifies the relation between "walking into" and "getting hurt." It would fire even in the situation "John walked into the mist" and lead to the inference "John got hurt." A complete encoding of the knowledge embodied in the rule would also specify the types/features of the argument fillers of "walk-into" for which the application of this rule would be appropriate. Given such an encoding, the propagation of binding from the first argument of *walk-into* to the argument of *hurt* will occur only if the fillers of the arguments of *walk-into* belong to the appropriate types (we discuss the encoding of such rules in sect. 5).

The use of categorization can also prevent certain cases of cross-talk in the representation of dynamic facts. For example, categorization may prevent cross-talk in the representation of *buy(Mary, Book1)* because spurious versions of this fact such as *buy(Book1, Mary)* would violate category restrictions and, hence, would be unstable. However, categorization cannot in and of itself solve the dynamic-binding problem, because it alone cannot enforce systematicity. For example, categorization cannot determine that the dynamic fact *give(John, Mary, Book1)* should result in the inferred fact *own(Mary, Book1)* but not *own(John, Book1)*.

2.5. The dynamic-binding problem in vision and language

The need for systematically dealing with composite objects in a dynamic manner immediately gives rise to the dynamic-binding problem. Thus the dynamic-binding problem occurs during any cognitive activity that admits systematicity and compositionality. Consider vision. Visual object recognition involves the rapid grouping of information over the spatial extent of an object and across different feature maps so that features belonging to one object are not confused with those of another (Treisman & Gelade 1980). The binding of features during visual processing is similar to the binding of argument fillers during reasoning. In terms of representational power, however, the grouping of all features belonging to the same object can be expressed using *unary*-predicates,[11] but as we have seen, reasoning requires the representation of unary as well as *n*-ary predicates. A similar need would arise in a more sophisticated vision system that dynamically represents and analyzes spatial relations between objects or parts of an object.

Although there may be considerable disagreement over the choice of primitives and the functional relationship between the "meaning" of a composite structure and that of its constituents, it seems apparent that a computational model of language should be capable of computing and representing composite structures in a systematic and dynamic manner. Thus language understanding requires a solution to the dynamic-binding problem, to support reasoning as well as syntactic processing and the dynamic linking of syntactic and semantic structures.

3. Solving the dynamic-binding problem

In this section we describe solutions to three technical problems associated with dynamic bindings discussed in sections 2.1 through 2.3. The solutions involve several ideas that complement each other and together lead to a connectionist model of knowledge representation and reflexive reasoning.

As pointed out in section 2.1, it is implausible to represent dynamic bindings by using structural changes, prewired interconnection networks, or by communicating names/pointers of arguments and fillers. Instead, what is required is a neurally plausible mechanism for rapidly and temporarily labeling the representations of fillers and predicate arguments to encode dynamically argument-filler bindings. Also required are mechanisms for systematically propagating such transient labels and allowing them to interact with long-term structures.

In the proposed system we use the temporal structure of node activity to provide the necessary labeling. Specifically, we represent dynamic bindings between arguments and fillers by the synchronous firing of appropriate nodes. We also propose appropriate representations for n-ary predicates, rules, long-term facts, and an *IS-A* hierarchy that facilitate the efficient propagation and recognition of dynamic bindings.[12]

The significance of temporally organized neural activity has long been recognized (Freeman 1981; Hebb 1949; Sejnowski 1981). In particular, von der Malsburg (1981; 1986) has proposed that correlated activity within a group of cells can be used to represent the dynamic grouping of cells. He also used temporal synchrony and synapses that can alter their weights within hundreds of milliseconds to model sensory segmentation and the human ability to attend to a specific speaker in a noisy environment (von der Malsburg & Schneider 1986). Abeles (1982; 1991) has put forth the hypothesis that computations in the cortex occur via "synfire chains" – propagation of synchronous activity along diverging and converging pathways between richly interconnected cell assemblies. Crick (1984) has also suggested that the use of fine temporal coincidence to represent dynamic bindings and synchronized activity across distant regions forms the keystone of Damasio's (1989) general framework for memory and consciousness. Several researchers have reported the occurrence of synchronous activity in the cat and monkey visual cortex and presented evidence in support of the conjecture that the visual cortex may be using synchronous and/or oscillatory activity to solve the binding problem (see sect. 7).

Recently, other researchers have used temporal synchrony to solve various aspects of the binding problem in visual perception (Horn et al. 1991; Hummel & Biederman 1992; Strong & Whitehead 1989). In this work we use temporal synchrony to solve a different problem, namely, the representation of, and systematic reasoning with, conceptual knowledge. In solving this problem we also demonstrate that temporal synchrony can support more complex representations. The expressiveness and inferential power of our model exceed that of the models cited above, because our system can represent dynamic instantiations of n-ary predicates, including multiple instantiations of the same predicate.[13]

Clossman (1988) has used synchronous activity to represent argument-filler bindings, but he has not suggested an effective representation of "rules" (and long-term facts). Consequently, his system could not propagate dynamic bindings to perform inferences.

As an abstract computational mechanism, temporal synchrony can be related to the notion of *marker passing* (Fahlman 1979; Quillian 1968).[14] Fahlman has proposed the design of a marker-passing machine (NETL) consisting of a parallel network of simple processors and a serial computer that controlled the operation of the parallel network. Each node could store a small number of discrete "markers" (or tags) and each link could propagate markers between nodes under the supervision of the network controller. Fahlman showed how his machine could compute transitive closure and set intersection in parallel, and in turn, solve a class of inheritance and recognition problems efficiently. Fahlman's system, however, was not neurally plausible. First, nodes in the system were required to store, match, and selectively propagate marker bits. Although units with the appropriate memory and processing characteristics may be readily realized, using electronic hardware, they do not have any direct neural analog. Second, the marker-passing system operated under the strict control of a serial computer that specified, "at every step of the propagation, exactly which types of links were to pass which markers in which directions" (Fahlman 1979).

The relation between marker passing and temporal synchrony can be recognized by noting that nodes firing in synchrony may be viewed as being marked with the *same* marker, and the propagation of synchronous activity along a chain of connected nodes can be viewed as the propagation of markers. Thus, in developing our reasoning system using temporal synchrony we have also established that marker-passing systems can be realized in a neurally plausible manner. In the proposed system, nothing has to be stored at a node in order to mark it with a marker. Instead, the time of firing of a node relative to other nodes and the coincidence between the time of firing of a node and that of other nodes has the effect of marking a node with a particular marker! A node in our system is not required to match any markers, it simply has to detect whether appropriate inputs are coincident. Our approach enables us to realize the abstract notion of markers by using time, a dimension that is forever present, and giving it added representational status.

As we shall see, the neural plausibility of our system also results from its ability to operate without a central controller. Once a query is posed to the system by activating appropriate nodes, it computes the solution without an external controller directing the activity of nodes at each step of processing (see also sect. 9.1).

Several other connectionist solutions to the binding problem have been suggested (Barnden & Srinivas 1991; Dolan & Smolensky 1989; Feldman 1982; Lange & Dyer 1989; Touretzky & Hinton 1988). These alternatives are discussed in section 9.3.

3.1. Representing dynamic bindings

Refer to the representation of some predicates and entities shown in Figure 2. Observe that predicates, their arguments, and entities are represented by using distinct nodes. For example, the ternary predicate *give* is represented by the three argument nodes labeled *giver*, *recip*, and *g-obj* together with an associated "node" depicted as a dotted rectangle (the role of the latter is specified in sect. 3.3). For simplicity we assume that each argument node corresponds to an individual connectionist node; this is an idealization. In section 7.3 we discuss how each argument node corresponds to an ensemble of nodes. Nodes such as *John* and *Mary* correspond to *focal* nodes of more elaborate connectionist representations of the entities "John" and "Mary." Information about the attribute values (features) of "John" and its relationship to other concepts is encoded by linking the focal node *John* to appropriate nodes. (Details of such an encoding may be found in Shastri 1991; Shastri & Feldman 1986). As explained by Feldman (1989), a focal node may also be realized by a small ensemble of nodes.

Dynamic bindings are represented in the system by the synchronous firing of appropriate nodes. Specifically, a dynamic binding between a predicate argument and its filler is represented by the synchronous firing of nodes that represent the argument and the filler. With reference to the nodes in Figure 2, the dynamic bindings (*giver* = *John*, *recipient* = *Mary*, *give-object* = *Bookl*) are represented by the rhythmic pattern of activity shown in Figure 3. These bindings encode the dynamic fact *give*(*John*, *Mary*, *Bookl*). The absolute phase of firing of filler and argument nodes is not significant – what matters is the coincidence (or the lack thereof) in the firing of nodes. The activity of the dotted rectangular nodes is not significant at this point and is not specified. As another example, consider the firing pattern shown in Figure 4. This pattern of activation represents the single binding (*giver* = *John*) and corresponds to the partially instantiated fact *give*(*John*,*x*,*y*), (i.e., "John gave someone something").

Figure 5 shows the firing pattern of nodes corresponding to the dynamic representation of the bindings (*giver* = *John*, *recipient* = *Mary*, *give-object* = *Bookl*, *owner* = *Mary*, *own-object* = *Bookl*, *potential-seller* = *Mary*, *can-sell-object* = *Bookl*). These bindings encode the facts *give*(*John*, *Mary*, *Bookl*), *own*(*Mary*, *Bookl*), and *can-sell*(*Mary*, *Bookl*). Observe that the (multiple) bindings between *Mary* and the arguments *recipient*, *owner*, and *potential-seller* are represented by these argument nodes firing in-phase with *Mary*. Further, the individual concepts *Mary*, *Bookl*, and *John* are firing out of phase and occupy distinct phases in the rhythmic pattern of activity.

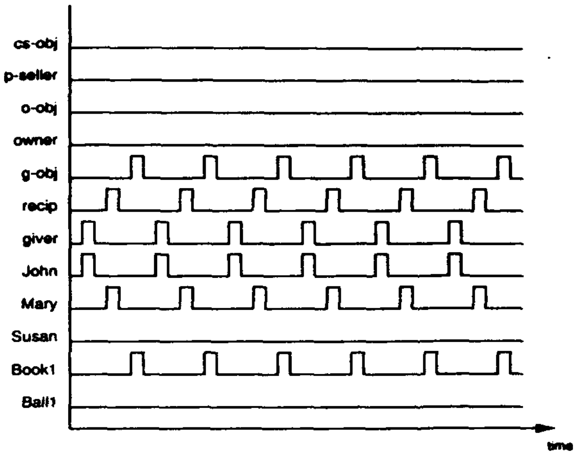


Figure 3. Rhythmic pattern of activation representing the dynamic bindings (*giver* = *John*, *recipient* = *Mary*, *give-object* = *Bookl*). These bindings constitute the fact *give*(*John*, *Mary*, *Bookl*). The binding between an argument and a filler is represented by the in-phase firing of associated nodes.

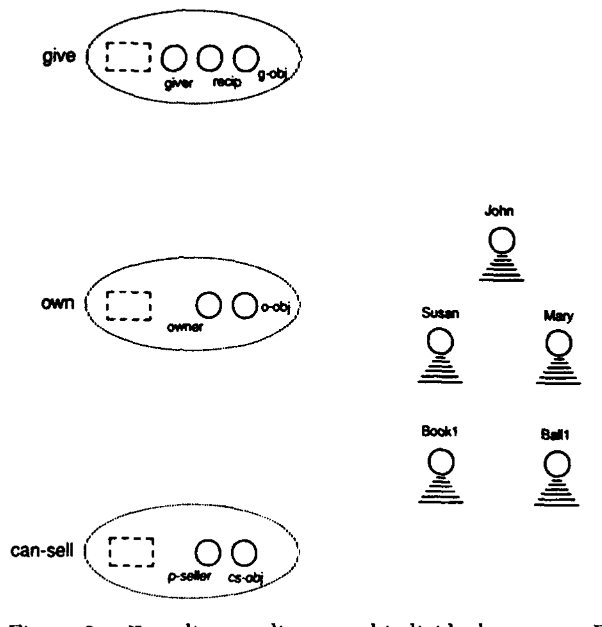


Figure 2. Encoding predicates and individual concepts: Distinct predicates and arguments are encoded using distinct nodes (in sect. 7.4 we discuss how nodes may be replaced by an ensemble of nodes). The hatched lines below concept nodes are intended to highlight that these nodes are just focal nodes of a much richer representation of concepts.

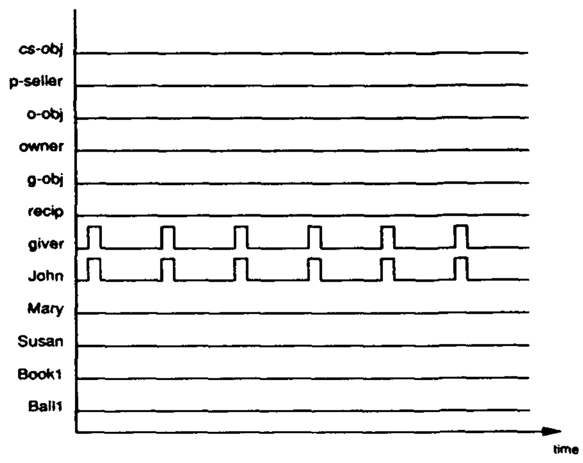


Figure 4. Representation of the dynamic binding (*giver* = *John*) that constitutes the partially instantiated fact "John gave someone something."

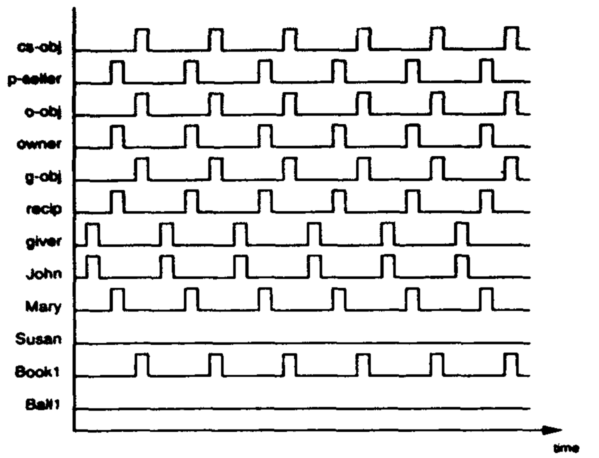


Figure 5. Pattern of activation representing the dynamic bindings (*giver = John, recipient = Mary, give-object = Book 1, owner = Mary, own-object = Book1, potential-seller = Mary, can-sell-object = Book1*). These bindings constitute the facts *give(John, Mary, Book1), own(Mary, Book1),* and *can-sell(Mary, Book1).* The transient representation of an entity is simply a phase within an oscillatory pattern of activity. The number of distinct phases required to represent a set of dynamic bindings equals only the number of distinct entities involved in the bindings. In this example three distinct phases are required. The bindings between *Mary* and the arguments *recipient, owner,* and *potential-seller* are represented by the in-phase firing of the appropriate argument nodes with *Mary.*

This highlights significant aspects of the proposed solution:

1. The transient or short-term representation of an entity is simply a *phase* within a rhythmic pattern of activity.

2. The number of *distinct* phases within the rhythmic activation pattern only equals the number of *distinct* entities participating in the dynamic bindings; this does not depend on the total number of dynamic bindings represented by the activation pattern.

3. The number of distinct entities that can participate in dynamic bindings at the same time is limited by the ratio of the period of the rhythmic activity and the width of individual spikes.

Thus far we have assumed that nodes firing in synchrony fire precisely in-phase. This is an idealization. In general we would assume a coarser form of synchrony, where nodes firing with a lag or lead of less than $\omega/2$ of one another would be considered to be firing in synchrony. This corresponds to treating the width of the "window of synchrony" to be ω.

3.2. Encoding rules and propagating dynamic bindings

In section 2.2 we described how a step of inference or rule application may be viewed as taking an instance of the antecedent predicate and dynamically creating an instance of the consequent predicate, with the argument bindings of the latter being determined by (1) the argument bindings of the former, and (2) the argument correspondence specified by the rule. Consequently, the encoding of a rule should provide a means for propagating bindings from the arguments of the antecedent predicate to the arguments of the consequent predicate in accordance with the argument correspondence specified in the rule. With reference to Figure 2, encoding the rules $\forall x,y,z\ [give(x,y,z) \Rightarrow own(y,z)]$ and $\forall u,v\ [own(u,v) \Rightarrow can\text{-}sell(u,v)]$ should have the following effect: The state of activation described by the rhythmic activation pattern shown in Figure 3 should eventually lead to the rhythmic activation pattern shown in Figure 5.

The desired behavior may be realized if a rule is encoded by linking the arguments of the antecedent and consequent predicates so as to reflect the correspondence between arguments specified by the rule. For example, the rule $\forall x,y,z\ [give(x,y,z) \Rightarrow own(y,z)]$ can be encoded by establishing links between the arguments *recipient* and *give-object* of *give* and the arguments *owner* and *own-object* of *own*, respectively. If we also wish to encode the rule $\forall x,y\ [buy(x,y) \Rightarrow own(x,y)]$, we can do so by connecting the arguments *buyer* and *buy-object* of *buy* to the arguments *owner* and *own-object* of *own*, respectively. This encoding is illustrated in Figure 6. In the idealized model we are assuming that each argument is represented as a single node and each argument correspondence is encoded by a one-to-one connection between the appropriate argument nodes. As discussed in section 7.3, however, each argument will be encoded as an ensemble of nodes and each argument correspondence will be encoded by many-to-many connections between the appropriate ensembles (for a preview see Fig. 26).

Arguments and concepts are encoded by using what we call ρ-btu nodes (where btu refers to "binary threshold unit"). These nodes have the following idealized behavior:

1. If a node A is connected to node B then the activity of node B will synchronize with the activity of node A. In particular, a periodic firing of A will lead to a periodic and in-phase firing of B. We assume that ρ-btu nodes can respond in this manner as long as the period of firing, π,

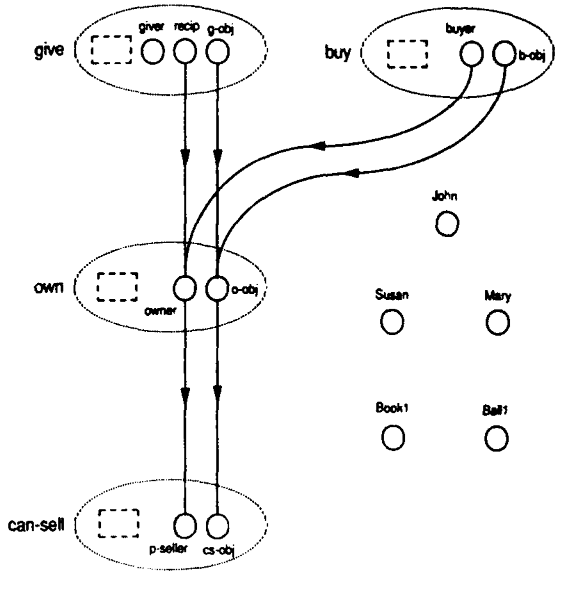


Figure 6. Encoding of predicates, individual concepts, and the rules $\forall x,y,z\ [give(x,y,z) \Rightarrow own(y,z)]$, $\forall x,y\ [own(x,y) \Rightarrow can\text{-}sell(x,y)]$, and $\forall x,y\ [buy(x,y) \Rightarrow own(x,y)]$. Links between arguments reflect the correspondence between arguments in the antecedents and consequents of rules.

lies in the interval [π_{min}, π_{max}]. This interval can be interpreted as defining the frequency range over which ρ-btu nodes can sustain a synchronized response.

2. To simplify the description of our model we will assume that periodic activity in a node can lead to synchronous periodic activity in a connected node within one period.

3. A threshold, *n*, associated with a node indicates that the node will fire only if it receives *n* or more synchronous inputs.[15] If unspecified, a node's threshold is assumed to be one.[16]

As described above, interconnected ρ-btu nodes can propagate synchronous activity and form chains of nodes firing in synchrony. In section 7 we point to evidence from neurophysiology and cite work on neural modeling that suggests that the propagation of synchronous activity is neurally plausible. Given the above interconnection pattern and node behavior, the initial state of activation shown in Figure 7 will lead to the state of activation shown in Figure 8 after one period, and to the state of activation shown in Figure 9 after another period.

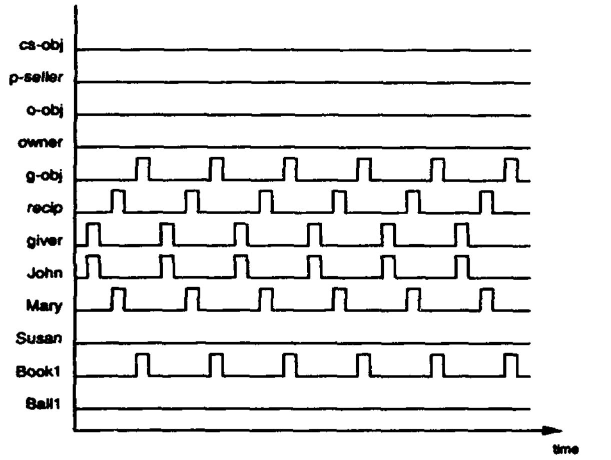


Figure 7. Initial pattern of activation representing the bindings (*giver = John, recipient = Mary, give-object = Book1*).

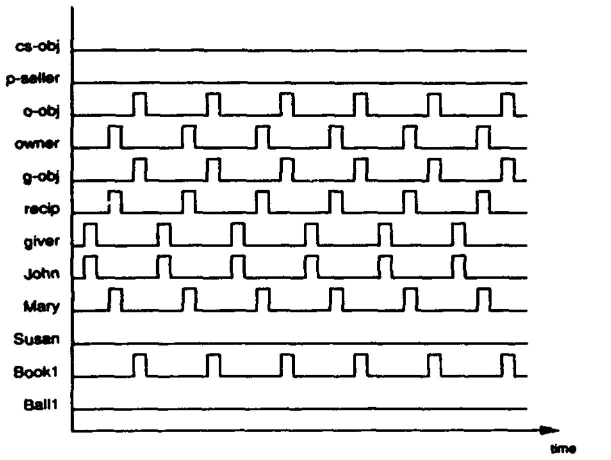


Figure 8. Pattern of activation after one period of oscillation (with reference to the state of activation in Figure 7). This state represents the dynamic bindings: (*giver = John, recipient = Mary, give-object = Book1, owner = Mary, own-object = Book1*). The system has essentially inferred the fact *own(Mary, Book1)*.

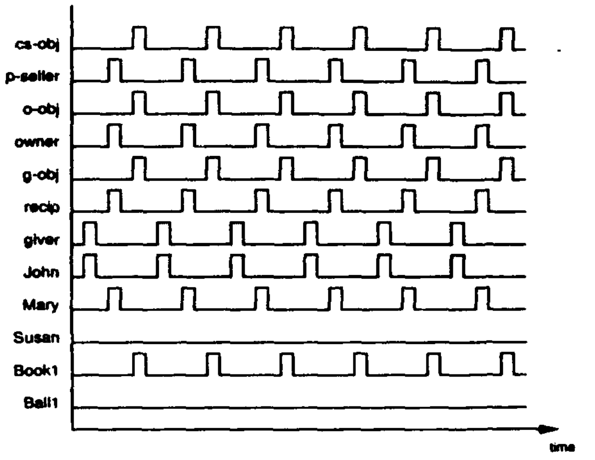


Figure 9. Pattern of activation after two periods of oscillation (with reference to the state of activation in Fig. 7). This state represents the dynamic bindings: (*giver = John, recipient = Mary, give-object = Book1, owner = Mary, own-object = Book1, potential-seller = Mary, can-sell-object = Book1*). The system has essentially inferred the facts *own(Mary, Book1)* and *can-sell(Mary, Book1)*.

The encoding of rules by the explicit encoding of the inferential dependency between predicates and predicate arguments, in conjunction with the use of temporal synchrony, provides an efficient mechanism for propagating dynamic bindings and performing systematic reasoning. Conceptually, the proposed encoding of rules creates a directed *inferential dependency graph:* Each predicate argument is represented by a node in the graph, and each rule is represented by links between nodes denoting the arguments of the antecedent and consequent predicates. In terms of this conceptualization, the evolution of the system's state of activity corresponds to a parallel breadth-first traversal of the directed inferential dependency graph. This means that (1) a large number of rules can fire in parallel, and (2) the time taken to generate a chain of inference is independent of the total number of rules and just equals $l\pi$ where l is the *length* of the chain of inference and π is the period of oscillatory activity.

3.3. Encoding long-term facts: Memory as a temporal pattern matcher

As stated in section 2.3, our system must also be capable of representing long-term facts, which are essentially a permanent record of a set of bindings describing a particular situation. The representation of a long-term fact should encode the bindings pertaining to the fact in a manner that allows the system to recognize rapidly dynamic bindings that match the encoded fact. Given that dynamic bindings are represented as temporal patterns, it follows that the encoding of a long-term fact should behave like a *temporal pattern matcher* that becomes active whenever the static bindings it encodes match the dynamic bindings represented in the system's state of activation.

The design of such a temporal pattern matcher is illustrated in Figures 10 and 11, which depict the encoding of the long-term facts *give(John, Mary, Book1)* and *give(John, Susan, x)*, respectively (the latter means "John

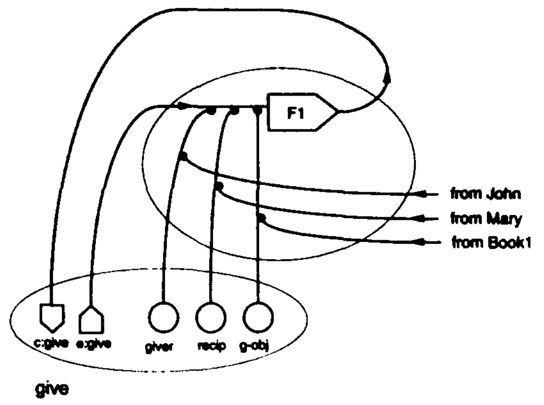


Figure 10. Encoding of a long-term fact: The interconnections shown here encode the static bindings (*giver-John, recipient = Mary, give-object = Book1*) that constitute the long-term fact *give(John, Mary, Book1)*. The pentagon-shaped nodes are τ-and nodes. A τ-and node becomes active if it receives an uninterrupted pulse train. The activation of *e:give* represents an externally or internally generated query asking whether the dynamic bindings indicated by the pattern of activity of argument nodes match the long-term knowledge encoded in the LTKB. The activation of *c:give* represents an assertion by the system that these dynamic bindings match the knowledge encoded in the LTKB.

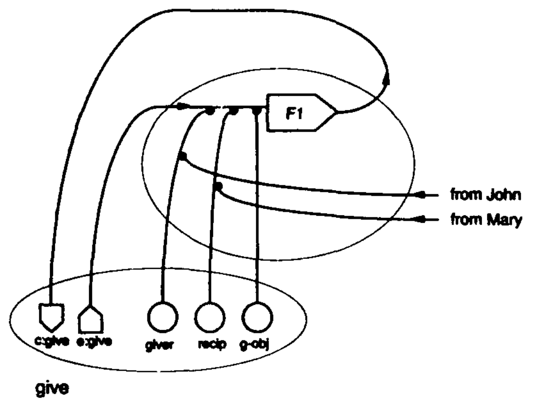


Figure 11. Encoding of the partially instantiated long-term fact *give(John, Mary, x)*, that is "John gave Mary something." The input from *g-obj* does not receive an inhibitory input from any filler.

gave Susan something"). The encoding fully specifies how a predicate is encoded. Observe that in addition to ρ-btu nodes, the encoding also makes use of pentagon shaped τ-and nodes that have the following idealized behavior:

1. A τ-and node becomes active on receiving an *uninterrupted* pulse train, that is, a pulse train such that the gap between adjacent pulses is less than a spike width. Thus a τ-and node behaves like a *temporal and* node. On becoming active, such τ node produces an output pulse train similar to the input pulse train.

2. Note that a τ-and node driven by a periodic input consisting of a train of pulses of width comparable to the period π, will produce a periodic train of pulses of width and periodicity π. We assume that a τ-and node can behave in this manner as long as the period of the input pulse train lies in the interval $[\pi_{min}, \pi_{max}]$.

3. A threshold, *n*, associated with a τ-and node indicates that the node will fire only if it receives *n* or more synchronous pulse trains. If unspecified, *n* is assumed to be one.

An *n*-ary predicate *P* is encoded by using two τ-and nodes and *n* ρ-btu nodes. One of these τ-and nodes is referred to as the *enabler* and the other as the *collector*. An enabler will be referred to as *e:P* and drawn pointing upward whereas a collector will be referred to as *c:P* and drawn pointing downward. With reference to Figures 10 and 11, the ternary predicate *give* is represented by the enabler *e:give*, the collector *c:give*, and the three argument nodes – *giver*, *recip*, and *g-obj*. The representational significance of the enabler and collector nodes is as follows. The enabler *e:P* of a predicate *P* has to be activated whenever the system is queried about *P*. Such a query may be posed by an external process or generated internally by the system itself during an episode of reasoning (see sect. 4.4). On the other hand, the system activates the collector *c:P* of a predicate *P* whenever the dynamic bindings of the arguments of *P* match the knowledge encoded in the LTKB.

A long-term fact is encoded using a τ-and node which receives an input from the enabler node of the associated predicate. This input is modified by inhibitory links from argument nodes of the associated predicate. If an argument is bound to an entity, the modifier input from the argument node is in turn modified by an inhibitory link from the appropriate entity node. The output of the τ-and node encoding a long-term fact is connected to the collector of the associated predicate. We will refer to the τ-and node associated with a long-term fact as a fact node. Note that there is only one enabler node, one collector node, and one set of argument nodes for each predicate. These nodes are shared by all the long-term facts pertaining to that predicate.

It can be shown that a fact node becomes active if and only if the static bindings it encodes match the dynamic bindings represented in the network's state of activation. As stated above, *e:P* becomes active whenever any query involving the predicate *P* is represented in the system. Once active, *e:P* outputs an uninterrupted pulse train that propagates to various fact nodes attached to *e:P*. Now the pulse train arriving at a fact node will be interrupted by an active argument of *P*, unless the filler of this argument specified by the long-term fact is firing in synchrony with the argument. But a filler and an argument will be firing in synchrony if and only if they are bound in the dynamic bindings. Thus a fact node will receive an uninterrupted pulse if and only if the dynamic bindings represented in the system's state of activation are such that either an argument is unbound, or if bound, the argument filler in the dynamic binding matches the argument filler specified in the long-term fact. The reader may wish to verify that the encodings given in Figures 10 and 11 will behave as expected.

The encoding of the long-term fact *give(John, Mary, Book1)* will recognize dynamic bindings that represent dynamic facts such as *give(John, Mary, Book1)*, *give (John, Mary, x)*, *give(x, Mary, y)*, and *give(x,y,z)*. However, it will not recognize those that represent *give(Mary, John, Book1)* or *give(John, Susan, x)*. Similarly, the encoding of the long-term fact *give(John, Susan, x)* will recognize dynamic bindings that encode *give(John, Susan, x)*, *give(x, Susan, y)*, and *give(x, y, z)*, but not *give(Susan, John, x)* or *give(John, Susan, Car7)*.

3.4. Dynamic bindings and temporal synchrony

Given the representation of dynamic bindings and the encoding of rules described in the preceding sections, one may view (1) reasoning as the transient but systematic propagation of a rhythmic pattern of activation, (2) an object in the dynamic memory as a phase in the above rhythmic activity, (3) bindings as the in-phase firing of argument and filler nodes, (4) rules as interconnection patterns that cause the propagation and transformation of such rhythmic patterns of activation, and (5) facts as temporal pattern matchers. During an episode of reasoning, all the arguments bound to a filler become active in the same phase as the filler, thereby creating transient "temporal frames" of knowledge grouped together by temporal synchrony. This can be contrasted with "static" frames of knowledge where knowledge is grouped together, spatially, using hard-wired links and nodes.

The system can represent a large number of dynamic bindings at the same time, provided the number of distinct entities involved in these bindings does not exceed $\lfloor \pi_{max}/\omega \rfloor$, where π_{max} is the maximum period (or the lowest frequency) at which ρ-btu nodes can sustain synchronous oscillations and ω is the width of the window of synchrony. Recall that a window of synchrony of ω implies that nodes firing with a lag or lead of less than ω/2 of one another are considered to be in synchrony. (We discuss biologically plausible values of π and ω in sect. 7.2 and the psychological implications of these limits in sect. 8.) As described thus far, the system allows the simultaneous representation of a large number of dynamic facts but only supports the representation of one dynamic fact per predicate. (In sect. 6 we discuss a generalization of the proposed representation that allows multiple dynamic facts pertaining to each predicate to be active simultaneously.)

Although synchronous activity is central to the representation and propagation of binding, the system does not require a global clock or a central controller. The propagation of in-phase activity occurs automatically – once the system's state of activation is initialized to represent an input situation by setting up appropriate dynamic bindings, the system state evolves automatically to represent the dynamic bindings corresponding to situations that follow from the input situation.

Reasoning is the spontaneous outcome of the system's behavior. The system does not encode syntactic rules of inference such as *modus ponens*. There is no separate interpreter or inference mechanism in the system that manipulates and rewrites symbols. The encoding of the LTKB is best viewed as a vivid internal model of the agent's environment. When the nodes in this model are activated to reflect a particular situation in the environment, the model simulates the behavior of the external world and dynamically creates a vivid model of the state of affairs resulting from the given situation. The system is clearly not a rule following system. At the same time it is not rule described or rule governed in the sense that a falling apple is. As Hatfield (1991) argues, the system is best described as being rule instantiating.

3.5. From mechanisms to systems

The mechanisms proposed in the previous sections provide the building blocks for a connectionist system that can represent and reason with knowledge involving *n*-ary predicates and variables. These mechanisms may interact in different ways to realize different sorts of reasoning behavior. For example, they can lead to a forward-reasoning system that can perform predictive inferences. Our discussion in the previous sections was in the context of such a system.

The proposed mechanisms may also be used to create a backward-reasoning system that behaves as follows: If the system's state of activation is initialized to represent a query, it attempts to answer the query based on the knowledge encoded in its LTKB. A backward-reasoning system may be generalized to perform explanatory inferences. If the state of such a system is initialized to represent an input "situation," it will automatically attempt to explain this situation on the basis of knowledge in its LTKB and a "minimal" set of assumptions.

With the aid of additional mechanisms it is possible to design a system that performs both predictive and explanatory inferences. Such a system would make predictions based on incoming information and at the same time seek explanations for, and test the consistency of, this information.

4. A backward-reasoning system

This section describes a backward-reasoning system based on the representational mechanisms described in section 3. The system encodes facts and rules in its LTKB and answers queries on the basis of this knowledge. For example, if the system encodes rules $\forall x,y,z\,[give(x,y,z) \Rightarrow own(y,z)]$ and $\forall u,v\,[own(u,v) \Rightarrow can\text{-}sell(u,v)]$, and the long-term fact "John bought Porsche7," it will respond yes to queries such as, Does John *own* Porsche7? or Can John *sell* something? The time taken to respond yes to a query is only proportional to the length of the shortest derivation of the query and is independent of the size of the LTKB.

In subsequent sections we describe several extensions of the backward-reasoning system. In section 5 we show how the system may be combined with an *IS-A* hierarchy that encodes entities, types (categories), and the super-/subconcept relations between them. The augmented system allows the occurrence of types, nonspecific instances of types, as well as entities in rules, facts, and queries. This in turn makes it easier to encode the appropriateness aspect of rules. An extension of the system to perform abduction is described in Ajjanagadde (1991).

4.1. The backward-reasoning system – a functional specification

The reasoning system can encode rules of the form:[17]

$$\forall x_1, \ldots, x_m\, [P_1(\ldots) \wedge P_2(\ldots) \ldots \wedge P_n(\ldots) \Rightarrow \exists z_1, \ldots z_l\, Q(\ldots)]$$

The arguments of P_is are elements of $\{x_1, x_2, \ldots x_m\}$. An argument of Q is either an element of $\{x_1, x_2, \ldots x_m\}$, or an element of $\{z_1, z_2, \ldots z_l\}$, or a constant. It is required that any variable occurring in multiple argument positions in the antecedent of a rule must also appear in the consequent.[18] The significance of this constraint is discussed in section 4.9. Additional examples of rules are:

$$\forall x,y,t\ [omnipresent(x) \Rightarrow present(x,y,t)]$$

Anyone who is omnipresent is present everywhere at all times;

$$\forall x,y\ [born(x,y) \Rightarrow \exists t\ present(x,y,t)]$$

Everyone must have been present at his or her birthplace sometime.

$$\forall x\ [triangle(x) \Rightarrow number\text{-}of\text{-}sides(x, 3)]$$

$$\forall x,y\ [sibling(x,y) \wedge born\text{-}together(x,y) \Rightarrow twins(x,y)]$$

Facts are assumed to be partial or complete instantiations of predicates. In other words, facts are atomic formulae of the form $P(t_1,t_2...t_k)$, where t_is are either constants or distinct existentially quantified variables. Some examples of facts are:

give(John, Mary, Book1);	John gave Mary Book1.
give(x, Susan, Ball2);	Someone gave Susan Ball2.
buy(John x);	John bought something.
own(Mary, Ball1);	Mary owns Ball1.
omnipresent(x);	There exists someone who is omnipresent.
triangle(A3);	A3 is a triangle.
sibling(Susan, Mary);	Susan and Mary are siblings.
born-together(Susan, Mary);	Susan and Mary were born at the same time.

A query has the same form as a fact. A query, all of whose arguments are bound to constants, corresponds to the yes-no question, "Does the query follow from rules and facts encoded in the long-term memory of the system?" A query with existentially quantified variables, however, has several interpretations. For example, the query $P(a,x)$, where a is a constant and x is an existentially quantified argument, may be viewed as the yes-no query: "Does $P(a,x)$ follow from the rules and facts for some value of x?" Alternatively this query may be viewed as the *wh*-query: "For what values of x does $P(a,x)$ follow from the rules and facts in the system's long-term memory?" Table 1 lists some queries, their interpretation(s), and their answer(s).

In describing the backward reasoner we begin by making several simplifying assumptions. We assume that rules have a single predicate in the antecedent and that constants and existentially quantified variables do not appear in the consequents of rules. We also restrict ourselves to yes-no queries at first. Subsequent sections will provide the relevant details.

Table 1. *Interpretation of some queries and their answers*

Query	yes-no form (answer)	*wh*-form (answer)
own(Mary,Ball1)	Does Mary own Ball1? (yes)	—
can-sell(Mary, Book1)	Can Mary sell Book1? (yes)	—
can-sell(Mary,x)	Can Mary sell something? (yes)	What can Mary sell? (Book1, Ball1)
own(x,y)	Does someone own something? (yes)	Who owns something? (Susan, Mary, John) What is owned by someone? (Book1, Ball1, Ball2)
can-sell(John,x)	Can John sell something? (yes)	What can John sell? (something, but don't know what)
present(x,North-pole,1/1/89)	Was someone present at northpole on 1/1/89? (yes)	Who was present at northpole on 1/1/89? (There was someone, but don't know who)
number-of-sides(A3,4)	Does A3 have 4 sides? (no)	—
can-sell(Mary, Ball2)	Can Mary sell Ball2? (no)	—
twins(Susan, Mary)	Are Mary and Susan twins? (yes)	—

4.2. Encoding rules and facts in long-term memory

Figure 12 depicts the encoding of the rules $\forall x,y,z$ $[give(x,y,z) \Rightarrow own(y,z)]$; $\forall x,y$ $[buy(x,y) \Rightarrow own(x,y)]$; and $\forall x,y$ $[own(x,y) \Rightarrow can\text{-}sell(x,y)]$, and the facts *give(John, Mary, Book1)*, *buy(John, x)*, and *own(Mary, Ball1)*.

As stated in section 3, a constant (i.e., an entity) is represented by a ρ-btu node, and an n-ary predicate is represented by a pair of τ-and nodes and n ρ-btu nodes. One of the τ-and nodes is referred to as the *enabler* and the other as the *collector*. An enabler is drawn pointing upward and is named *e:[predicate-name]*. A collector is drawn pointing downwards and is named *c:[predicate-name]*. The enabler, *e:P*, of a predicate *P* has to be activated whenever the system is queried about *P*. As we shall see, such a query may be posed by an external process or generated internally by the system during an episode of reasoning. On the other hand, the system activates the collector, *c:P*, of a predicate *P* whenever the current dynamic bindings of the arguments of *P* match the long-term knowledge encoded in the system. Each fact is encoded using a distinct τ-and node that is interconnected with appropriate enabler, collector, argument, and entity nodes (see sect. 3.3).

A rule is encoded by connecting (1) the collector of the antecedent predicate to the collector of the consequent

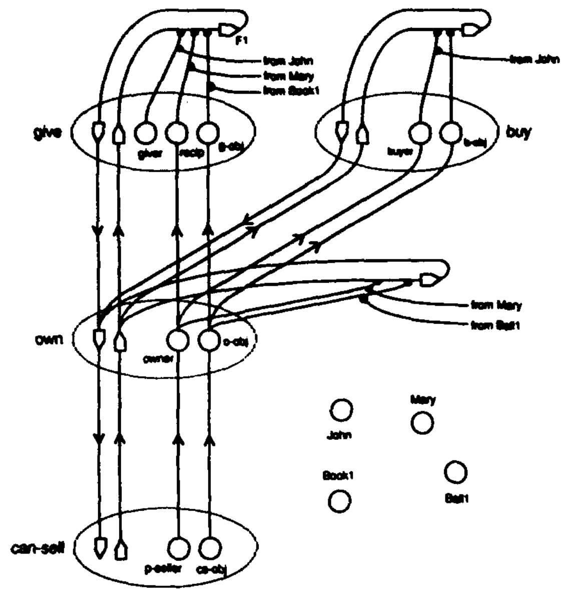


Figure 12. A network encoding the rules $\forall x,y,z$ *[give(x, y, z) ⇒ own(y, z)]*, $\forall x,y$ *[buy(x, y) ⇒ own(x, y)]*, and $\forall x,y$ *[own(x, y) ⇒ can-sell(x, y)]*; and the long-term facts *give(John, Mary, Book1), buy(John, x),* and *own(Mary, Book2).* The links between arguments are in the reverse direction because the rules are wired for "backward reasoning."

predicate, (2) the enabler of the consequent predicate to the enabler of the antecedent predicate, and (3) the argument nodes of the consequent predicate to the argument nodes of the antecedent predicate in accordance with the correspondence between these arguments specified in the rule (see Fig. 12). Notice that the links are directed from the arguments of the consequent predicate to the arguments of the antecedent predicate. The direction of links is reversed because the system performs backward reasoning.

4.3. Posing a query: Specifying dynamic bindings

A query is a (partially) specified predicate instance of the form $P(t_1, \ldots, t_n)$?, where t_is are either constants (entities) or existentially quantified variables. Therefore, posing a query to the system involves specifying the query predicate and the argument bindings specified in the query. We will assume that only one external process communicates with the reasoning system. The possibility of communication among several modules is discussed in section 10.4 (also see sects. 10.1–10.3). Let us choose an arbitrary point in time – say, t_0 – as our point of reference for initiating the query. The argument bindings specified in the query are communicated to the network as follows:

1. Let the argument bindings involve m distinct entities: $c_1, \ldots, c_m$. With each c_i, associate a delay δ_i such that no two delays are within ω of one another and the longest delay is less than $\pi - \omega$. Here ω is the width of the window of synchrony, and π lies in the interval $[\pi_{min}, \pi_{max}]$.

2. The argument bindings of an entity c_i are indicated to the system by providing an oscillatory spike train of periodicity π starting at $t_0 + \delta_i$, to c_i and all arguments of the query predicate bound to c_i. As a result, a distinct phase is associated with each distinct entity introduced in the query and argument bindings are represented by the synchronous activation of the appropriate entity and argument nodes.

3. The query predicate is specified by activating *e:P*, the enabler of the query predicate *P*, with a pulse train of width and periodicity π starting at time t_0.

Observe that posing a query simply involves activating the enabler node of the query predicate and the arguments and fillers specified in the query. There is no central controller that monitors and regulates the behavior of individual nodes at each step of processing.

4.4. The inference process for yes-no queries

Once a query is posed to the system, its state of activation evolves automatically and produces an answer to the query. The activation of the collector node of the query predicate indicates that the answer to the query is yes. The time taken by the system to produce a yes answer equals $2\pi(l + 1)$, where π is the period of oscillation of nodes and l equals the length of the shortest derivation of the query.[19] If the collector node of the query predicate does not receive any activation within $2(d + 1)$ periods of oscillations, where d equals the diame[illegible] the inferential dependency graph, the answer to [illegible]ry is "don't know." If we make the closed-world ass[illegible]on,[20] then a don't-know answer can be viewed as a no answer.

We illustrate the inference process with the help of an example (see Fig. 12). Consider the query *can-sell(Mary, Book1)*? (i.e., Can Mary sell Book1?). This query is posed by providing inputs to the entities *Mary* and *Book1*, the arguments *p-seller, cs-obj*, and the enabler *e:can-sell*, as shown in Figure 13. Observe that *Mary* and *p-seller*

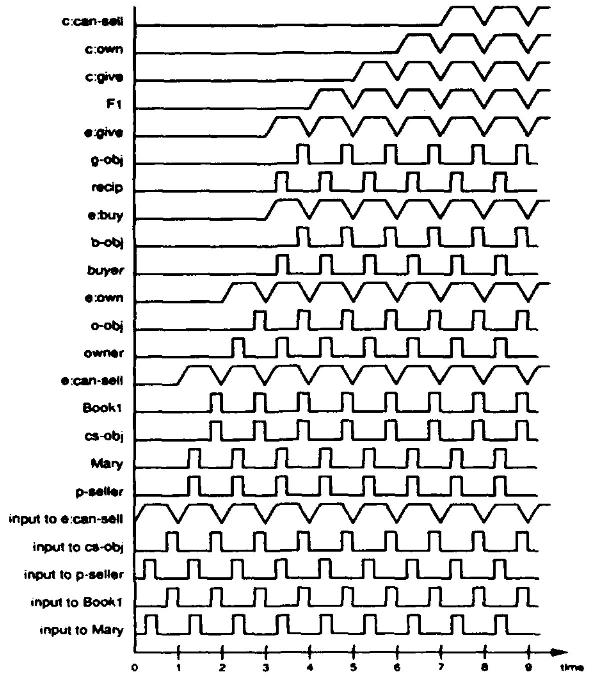


Figure 13. Activation trace for the query *can-sell(Mary, Book1)*? (Can Mary sell Book1?). The query is posed by providing an oscillatory input to *e:can-sell, Mary, Book1, p-seller*, and *cs-obj* as shown. The activation of *c:can-sell* indicates a yes answer.

receive synchronous activation and so do *Book1* and *cs-obj*. Let us refer to the phase of activity of *Mary* and *Book1* as phase-1 and phase-2, respectively.

As a result of the inputs, *Mary* and *p-seller* fire synchronously in phase-1, whereas *Book1* and *cs-obj* fire synchronously in phase-2 of every period of oscillation. The node *e:can-sell* also oscillates and generates a pulse train of periodicity and pulse width π. The activations from the arguments *p-seller* and *cs-obj* reach the arguments *owner* and *o-obj* of the predicate *own*, and consequently, starting with the second period of oscillation, *owner* and *o-obj* become active in phase-1 and phase-2, respectively. Thereafter, the nodes *Mary, owner,* and *p-seller* are active in phase-1, whereas the nodes *Book1, cs-obj,* and *o-obj* are active in phase-2. At the same time, the activation from *e:can-sell* activates *e:own*. At this point the system has essentially created two dynamic bindings – *owner* = *Mary* and *own-object* = *Book1*. Given that *e:own* is also active, the system's state of activity now encodes the internally generated query *own(Mary, Book1)*? (i.e., Does Mary own Book1?).

The fact node associated with the fact *own(Mary, Ball1)* does not match the query and remains inactive. Recall that fact nodes are τ-and nodes and hence become active only upon receiving an uninterrupted pulse train (see sect. 3.3). Since *Ball1* is not firing, the inhibitory activation from the argument node *owner* interrupts the activation going from *e:own* to the fact node and prevents it from becoming active.

The activation from *owner* and *o-obj* reaches the arguments *recip* and *g-obj* of *give*, and *buyer* and *b-obj* of *buy*, respectively. Thus beginning with the third period, arguments *recip* and *buyer* become active in phase-1, whereas arguments *g-obj* and *b-obj* become active in phase-2. In essence, the system has created new bindings for the arguments of predicates *can-sell* and *buy*. Given that the nodes *e:buy* and *e:give* are also active, the system's state of activity now encodes two additional queries: *give(x, Mary, Book1)*? and *buy(Mary, Book1)*?.

The fact node representing the fact *buy(John, x)* does not become active because the activation from *e:buy* is interrupted by the inhibitory activations from the arguments *buyer* and *b-obj*. (Notice that *John* is not active.) The fact node *F1*, associated with the fact *give(John, Mary, Book1)* however, does become active as a result of the uninterrupted activation it receives from *e:give*. Observe that the argument *giver* is not firing and the inhibitory inputs from the arguments recip and *g-obj* are blocked by the synchronous inputs from *Mary* and *Book1*, respectively. The activation from *F1* causes *c:give* to become active, and the output from *c:give* in turn causes *c:own* to become active and transmit an output to *c:can-sell*. Consequently *c:can-sell*, the collector of the query predicate *can-sell*, becomes active, resulting in an affirmative answer to the query.

4.5. Encoding rules with constants and repeated variables in the consequent

In this section we describe how rules containing constants (entities) and/or existentially quantified variables in the consequent are encoded. Consider the rule:

$$\forall x1,x2,y\ [P(x1,x2) \Rightarrow \exists z\ Q(x1,x2,y,z,a)] \qquad (3)$$

The encoding of rule (3) is shown in Figure 14. It uses a new type of node, which we refer to as a τ-or node (node *g1* in Fig. 14). Such a node behaves like a temporal *or* node and becomes active on receiving any input above its threshold and generates an oscillatory response with a period and pulse width equal to π_{max}, the maximum period at which the ρ-btu nodes can sustain synchronous activity.

Node *g1* projects inhibitory modifiers to links between argument and enabler nodes that can block the firing of the rule. The node *g1* ensures that the rule participates in an inference only if all the conditions implicit in the consequent of the rule are met. The first condition concerns the occurrence of existentially quantified variables in the consequent of a rule. Observe that such a rule only warrants the inference that there exist *some* filler of an existentially quantified argument and, hence, cannot be used to infer that a specific entity fills such an argument. Therefore, if an existentially quantified variable in the consequent of a rule gets bound in the reasoning process, the rule cannot be used to infer the consequent. With reference to rule (3), the desired behavior is achieved by the link from the existentially quantified (fourth) argument of *Q* to *g1* and the inhibitory modifiers emanating from *g1*. The node *g1* will become active and block the firing of the rule whenever the fourth argument of *Q* gets bound to any filler.

The second condition concerns the occurrence of entities in the consequent of a rule. Rule (3) cannot be used if its fifth argument is bound to any entity other than *a*. In general, a rule that has an entity in its consequent cannot be used if the corresponding argument gets bound to any other entity during the reasoning process. In the encoding of rule (3), this constraint is encoded by link from the fifth argument of *Q* to *g1* that is in turn modified by an inhibitory modifier from *a*. If the fifth argument of *Q* gets bound to any entity other than *a*, *g1* will become active and block the firing of the rule.

If the same variable occurs in multiple argument positions in the consequent of a rule, it means that this variable should either remain unbound or get bound to the same entity. This constraint can be encoded by introducing a node that receives

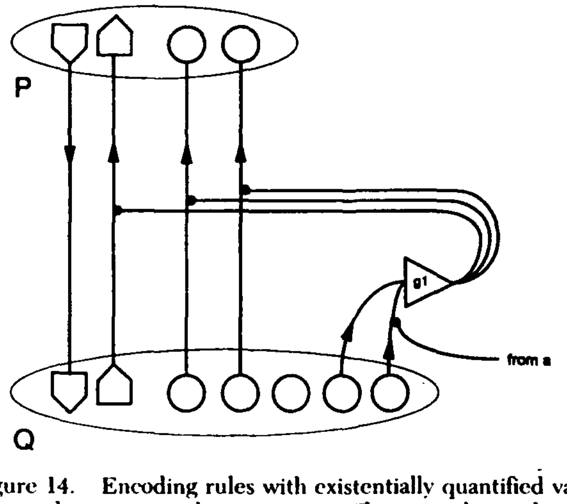


Figure 14. Encoding rules with existentially quantified variables and constants in the consequent: The network encodes the rule $\forall x1,x2,y\ [P(x1,x2) \Rightarrow \exists z\ Q(x1,x2,y,z,a)]$. This rule must not fire during the processing of a query, if either the existentially bound argument *z* gets bound, or the last argument gets bound to a constant other than *a*. The node *g1* is a τ-or node. It projects inhibitory modifiers that block the firing of the rule if the above condition is violated.

inputs from all the arguments that correspond to the same variable, and becomes active and inhibits the firing of the rule unless all such arguments are firing in synchrony. Observe that due to the temporal encoding, arguments bound to the same entity will fire in the same phase and, hence, a node need only check that the inputs from appropriate argument nodes are in synchrony to determine that the arguments are bound to the same entity. Consider the network fragment, shown in Figure 15, that depicts the encoding of the rule $\forall x\, P(x) \Rightarrow Q(x,x,a)$. The node *g2* is like a τ-or node except that it becomes active if it receives inputs in more than one phase within a period of oscillation. This behavior ensures that the firing of the rule is inhibited unless the appropriate arguments are bound to the same entity.

4.6. Encoding multiple antecedent rules

A rule with conjunctive predicates in the antecedent, that is, a rule of the form $P_1(\ldots) \wedge P_2(\ldots) \wedge \ldots P_m(\ldots) \Rightarrow Q(\ldots)$, is encoded using an additional τ-and node that has a threshold of *m*. The outputs of the collectors of $P_1, \ldots, P_m$ are connected to this node, which in turn is connected to the collector of *Q*. This additional node becomes active if and only if it receives inputs from the collector nodes of all the *m* antecedent predicates. The interconnections between the argument nodes of the antecedent and consequent predicates remain unchanged. Figure 16 illustrates the encoding of the multiple antecedent rule $\forall x,y P(x,y) \wedge Q(y,x) \Rightarrow R(x,y)$. The τ-and node labeled *g3* has a threshold of 2.

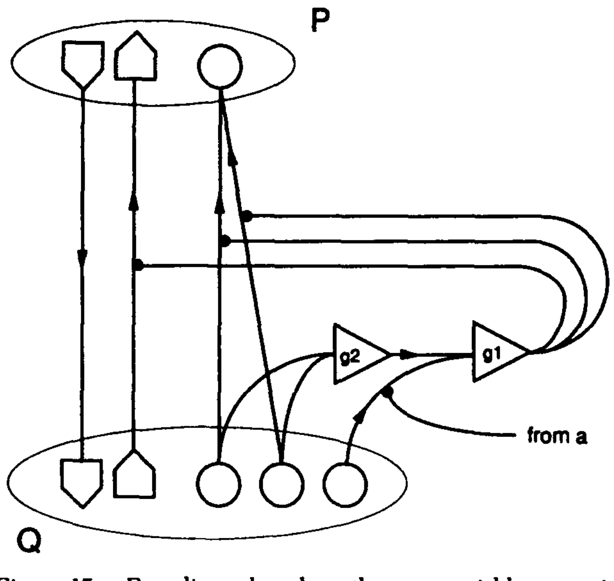


Figure 15. Encoding rules where the same variable occurs in multiple argument positions in the consequent: The network encodes the rule $\forall x\, P(x) \Rightarrow Q(x, x, a)$. The rule must fire only if a multiply occurring variable is unbound, or all occurrences of the variable are bound to the same constant. The node *g2* is like a τ-or node except that it becomes active if it receives inputs in more than one phase within a period of oscillation. On becoming active it activates the τ-or node *g1*. The firing of *g1* blocks the firing of the rule whenever the first and second arguments of *Q* get bound to different constants. (The encoding also enforces the constraint that the last argument of *Q* should not be bound to any constant other than *a*.)

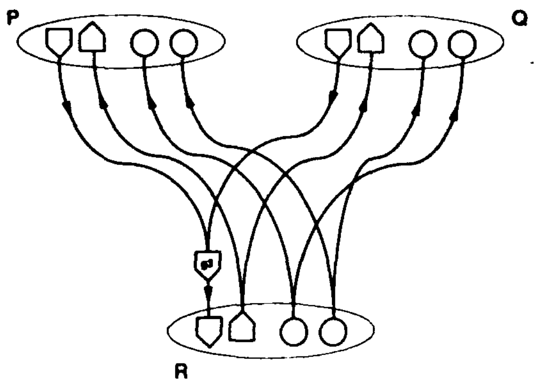


Figure 16. The encoding of the rule $\forall x,y P(x, y) \wedge Q(y, x) \Rightarrow R(x, y)$. The τ-and node labeled *g3* has a threshold of 2. Multiple antecedent rules are encoded by using an additional τ-and node whose threshold equals the number of predicates in the antecedent. This node becomes active on receiving inputs from the collector nodes of all the antecedent predicates.

4.7. Answering wh-queries

As stated in section 4.1, a query with unbound arguments can be interpreted either as a yes-no query or a *wh*-query. To answer a yes-no query the system need only determine whether there exist some instantiations of the unbound arguments. To answer a *wh*-query, however, the system must also determine the instantiations of unbound arguments for which the query is true. We describe how the proposed system can be extended to do so.

Consider the proof of the query *can-sell(Mary,x)?* with respect to the network shown in Figure 12. The yes-no version of this query will be answered in the affirmative and the two relevant facts *own(Mary, Ball1)* and *give(John, Mary, Book1)* will become active. The answer to the *wh*-query What can Mary sell? simply consists of the entities bound to the arguments *g-obj* and *b-obj*, respectively, of the two active facts. Observe that the arguments *g-obj* and *b-obj* are precisely the arguments that map to the unbound argument *cs-obj* of *can-sell* via the rules encoded in the system. The system can extract this information by the same binding propagation mechanism it uses to map bound arguments. A straightforward way of doing so is to posit an answer extraction stage that occurs after the yes-no query associated with the *wh*-query has produced a yes answer. For example, given the query What can Mary sell? the system first computes the answer to the yes-no query Can Mary sell something? and activates the facts that lead to a yes answer, namely, *own(Mary, Ball1)* and *give(John, Mary, Book1)*. The answer extraction stage follows and picks out the entities *Ball1* and *Book1* as the answers.

In order to support answer extraction, the representation of a fact is augmented as shown in Figure 17. The representation of a fact involving an *n*-ary predicate is modified to include *n* + 1 additional nodes: For each of the *n* arguments there is a two-input ρ-btu node with a threshold of 2. We refer to such a node as a *binder* node. The other node (shown as a filled-in pointed pentagon) is like a τ-and node except that once activated, it remains so even after the inputs are withdrawn (for several periods of

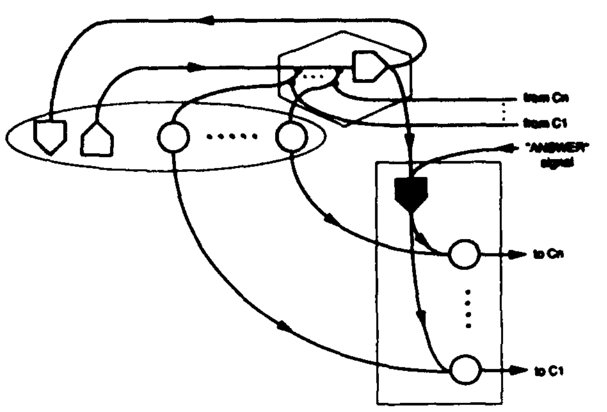


Figure 17. Augmented representation of a long-term fact in order to support answer extraction. For each argument of the associated predicate there exists a ρ-btu node with a threshold of 2. The node shown as a filled-in pentagon behaves like a τ-and node except that once activated, it stays active for some time – say about 20π – even after the inputs are withdrawn.

oscillations). This node, which we will refer to as a latch node, receives an ANSWER input and an input from the associated fact node.

At the end of the first stage, the fact nodes corresponding to all the relevant facts would have become active. The output of these nodes in conjunction with the ANSWER signal will turn on the associated latch nodes and provide one of the two inputs to the binder nodes. If the associated yes-no query results in a yes answer (i.e., the collector of the query predicate becomes active), the desired unbound arguments of the query predicate are activated in a distinct phase. The activation of these arguments eventually leads to the activation of the appropriate arguments in the facts relevant to answering the query. This provides an input to the appropriate binder nodes of these facts. Because the binder nodes were already receiving an input from a latch node, they become active and produce an output that activates the associated entities in phase with the appropriate query arguments. The answer to the *wh*-query (i.e., the entities that fill the argument a_i of the query) will be precisely those entities that are active in phase with a_i. The time taken by the answer extraction step is bounded by the depth of the inferential dependency graph.

4.8. Admitting function terms

The expressiveness and reasoning power of the system can be extended by allowing restricted occurrences of function terms in rules and facts. Function terms introduce new entities during the reasoning process. But given that entities are represented as a phase in the pattern of activity, an entity introduced by a function term can be represented by an additional phase in the rhythmic activity. Thus the reference to "*mother-of*(Tom)" during an episode of reasoning should lead to activity in a distinct phase. This phase would represent the "mother of Tom," and any arguments bound to the "mother of Tom" would now fire in this phase. A provisional solution along these lines is described by Ajjanagadde (1990).

4.9. Constraints on the form of rules

The encoding of rules described thus far enforces (1) the correspondence between the arguments of the antecedent and consequent predicates in a rule, and (2) equality among arguments in the consequent of a rule. In certain cases, however, it is difficult for the backward-reasoning system to enforce equality among arguments in the antecedent of a rule. Consider the rule $\forall x,y\ P(x,x,y) \Rightarrow Q(y)$ and the query $Q(a)$?. The processing of this query will result in the dynamic query $P(?,?,a)$? – where the first and second arguments will be left unspecified. Consequently, the system cannot enforce the condition implicit in the rule that a long-term fact involving P should match the query $Q(a)$ only if its first and second arguments are bound to the same constant. Performing such an equality test is complicated in a system that allows multiple predicates in the antecedent of rules and the chaining of inference. Consider the rule $\forall x,y\ P(x,y) \land R(x,y) \Rightarrow Q(y)$ and the query $Q(a)$?. The predicates P and R may be derivable from other predicates by a long sequence of rule application. Hence to derive the query $Q(a)$? the system may have to test the equality of arbitrary pairs of argument fillers in a potentially large number of facts distributed across the LTKB. It is conjectured that nonlocal and exhaustive equality testing cannot be done effectively in any model that uses only a linear number of nodes in the size of the LTKB and time that is independent of the size of the LTKB.

Contrast the situation described above with one wherein the rule is $\forall x,y\ P(x,x,y) \Rightarrow Q(x)$ and the query is $Q(a)$?. The dynamic query resulting from the processing of the query $Q(a)$? will be $P(a,a,y)$?. Notice that now the condition that the first and second arguments of P should be the same is automatically enforced by the propagation of bindings and is expressed in the dynamically generated query at P. The crucial feature of the second situation is that x, the *repeated* variable in the antecedent of the rule, also appears in the consequent *and* gets bound in the reasoning process. Thus, for the system to respond to a query, any variable occurring in multiple argument positions in the antecedent of a rule that participates in the answering of the query should also appear in the consequent of the rule and get bound during the query-answering process. This constraint is required in a backward-reasoning system but not in a forward-reasoning system. In the latter, the rule $\forall x,y,z\ P(x,y) \land Q(y,z) \Rightarrow R(x,z)$ would be encoded as shown in Figure 18. The τ-or node with a threshold of 2 receives inputs from the two argument nodes that should be bound to the same filler. It becomes active if it receives two inputs in the same phase and enables the firing of the rule via intermediary ρ-btu and τ-and nodes. This ensures that the rule fires only if the second and first arguments of P and Q, respectively, are bound to the same filler. In the case of forward reasoning, a rule that has variables occurring in multiple argument positions in its consequent can participate in the reasoning process provided such variables also appear in its antecedent and get bound during the reasoning process. The restrictions mentioned above on the form of rules exclude certain inferences (we discuss these exclusions and their implications in sect. 8.2).

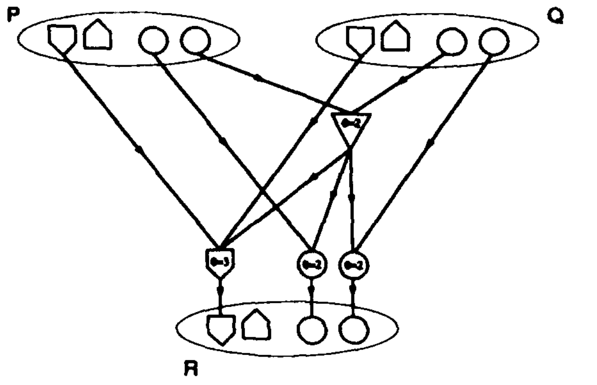


Figure 18. Encoding a rule with repeated variables in the antecedent within a forward reasoning system. The figure shows the encoding of the rule $\forall x,y,z\ P(x, y) \wedge Q(y, z) \Rightarrow R(x, z)$. This rule should fire only if the two arguments in the antecedent corresponding to variable y get bound to the same constant. The τ-or node with a threshold of 2 receives inputs from the two argument nodes that should be bound to the same filler. It becomes active if it receives two inputs in the same phase and enables the firing of the rule via intermediary ρ-btu and τ-and nodes. These nodes have suitable thresholds.

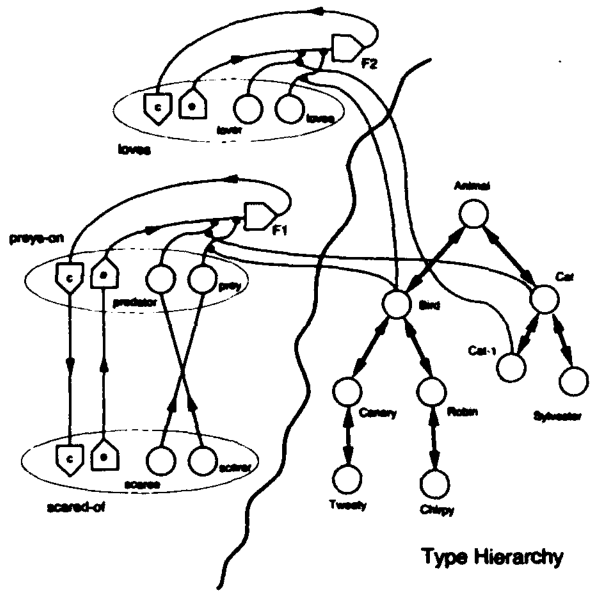


Figure 19. Interaction between a rule-based reasoner and an *IS-A* hierarchy. The rule component encodes the rule $\forall x,y$ *preys-on*$(x, y) \Rightarrow$ *scared-of*(y, x) and the facts $\forall x{:}Cat,\ y{:}Bird$ *preys-on*(x, y), and $\exists x{:}Cat\ \forall y{:}Bird$ *loves*(x, y). The first fact is equivalent to *preys-on(Cat, Bird)* and states that cats prey on birds. The second fact states that there is a cat that loves all birds.

5. Integrating the rule-based reasoner with an *IS-A* hierarchy

The rule-based reasoner described in the previous section can be integrated with an *IS-A* hierarchy representing entities, types (categories), the instance-of relations between entities and types, and the super-/subconcept relations between types. For convenience, we will refer to the instance-of, superconcept, and subconcept relations collectively as the *IS-A* relation. The augmented system allows the occurrence of types as well as entities in rules, facts, and queries. Consequently, the system can store and retrieve long-term facts such as "Cats prey on birds" and "John bought a Porsche" that refer to types (Cat and Bird) as well as nonspecific instances of types (a Porsche). The system can also combine rule-based reasoning with type inheritance. For example, it can infer "John owns a car" and "Tweety is scared of Sylvester" (the latter assumes the existence of the rule "If x preys-on y then y is scared of x" and the *IS-A* relations "Sylvester is a Cat" and "Tweety is a Bird"). Combining the reasoning system with an *IS-A* hierarchy also facilitates the representation of the appropriateness aspect of a rule. Recall that appropriateness concerns the applicability of the systematic aspect of a rule in a given situation, depending on the types of argument fillers involved in that situation. As we shall see, the augmented system allows knowledge in the *IS-A* hierarchy to interact with the encoding of the systematic aspects of a rule in order to enforce type restrictions and type preferences on argument fillers.

The integration of the reasoner with the *IS-A* hierarchy described below is a first cut at enriching the representation of rules. We only model the instance-of, subconcept, and superconcept relations and suppress several issues such as a richer notion of semantic distance, frequency- and category-size effects, and prototypicality (e.g., see Lakoff 1987).

Figure 19 provides an overview of the encoding and reasoning in the integrated reasoning system. The rule-base part of the network in Figure 19 encodes the rule $\forall x,y$ *[preys-on*$(x,y) \Rightarrow$ *scared-of*$(y,x)]$, and the facts $\forall x{:}Cat$, $y{:}Bird$ *preys-on*(x,y) and $\exists x{:}Cat$, $\forall y{:}Bird$ *loves*(x,y).

The first fact says "cats prey on birds" and is equivalent to *preys-on(Cat, Bird)*. The second fact states "there exists a cat that loves all birds." The type hierarchy in Figure 19 encodes the *IS-A* relationships: *is-a(Bird, Animal), is-a(Cat, Animal), is-a(Robin, Bird), is-a(Canary, Bird), is-a(Tweety, Canary), is-a(Chirpy, Robin)*, and *is-a(Sylvester, Cat)*. Facts involving typed variables are encoded in the following manner: A typed, universally quantified variable is treated as being equivalent to its type. Thus $\forall x{:}Cat,\ y{:}Bird$ *preys-on*(x,y) is encoded as *preys-on(Cat, Bird)*. A typed, existentially quantified variable is encoded using a unique subconcept of the associated type. Thus in Figure 19, $\exists x{:}Cat\ \forall y{:}Bird$ *loves*(x,y) is encoded as *loves(Cat-1, Bird)*, where *Cat-1* is some unique instance of *Cat*. In its current form, the system only deals with facts and queries wherein all existential quantifiers occur outside the scope of universal quantifiers.

For now let us assume that (1) each concept[21] (type or entity) in the *IS-A* hierarchy is encoded as a ρ-btu node, (2) each *IS-A* relationship, say *is-a(A, B)*, is encoded using two links – a bottom-up link from *A* to *B* and a top-down link from *B* to *A*, and (3) the top-down and bottom-up links can be enabled selectively by built-in, automatic control mechanisms. How this is realized is explained in section 5.2.

The time course of activation for the query *scared-of(Tweety, Sylvester)?* (Is Tweety scared of Sylvester?) is given in Figure 20. The query is posed by turning on *e:scared-of* and activating the nodes *Tweety* and *Sylvester* in synchrony with the first and second arguments of *scared-of*, respectively. The bottom-up links emanating

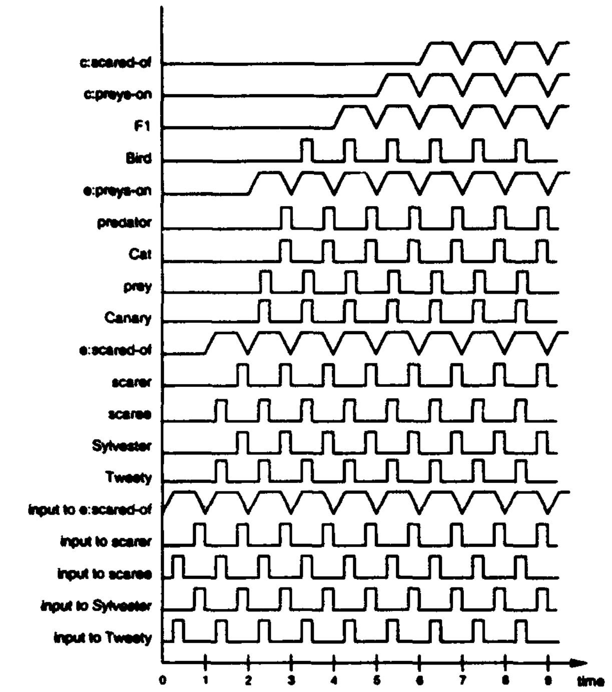


Figure 20. Activation trace for the query *scared-of(Tweety, Sylvester)?*, (i.e., Is Tweety scared of Sylvester?).

from *Tweety* and *Sylvester* are also enabled. The activation spreads along the *IS-A* hierarchy, and eventually *Bird* and *Cat* start firing in synchrony with *Tweety* and *Sylvester*, respectively. At the same time, the activation propagates in the rule base. As a result, the initial query *scared-of(Tweety, Sylvester)?* is transformed into the query *preys-on(Cat, Bird)?*, which matches the stored fact *preys-on(Cat, Bird)* and leads to the activation of *c:preys-on*. In turn *c:scared-of* becomes active and signals an affirmative answer.

There are advantages to expressing certain rules as facts. Although the reasoning system described in section 4 can use rules to draw inferences, it cannot retrieve the rules per se; for knowledge to be retrievable, it must be in the form of a fact. Hence integrating the rule-based reasoner with an *IS-A* hierarchy has added significance, because it allows certain rulelike knowledge to be expressed as facts, thereby making it retrievable in addition to being usable during inference. Consider "Cats prey on birds." The rule-based reasoner can only express this as the rule $\forall x, y\ Cat(x) \wedge Bird(y) \Rightarrow preys\text{-}on(x,y)$ and use it to answer queries such as *preys-on(Sylvester, Tweety)?*. It, however, cannot answer queries such as *preys-on(Cat, Bird)?* that can be answered by the integrated system.

5.1. Some technical problems

Two technical problems must be solved in order to integrate the *IS-A* hierarchy and the rule-based component. First, the encoding of the *IS-A* hierarchy should be capable of representing multiple instantiations of a concept. For example, in the query discussed above, the concept *animal* would receive activation originating at *Tweety* as well as *Sylvester*. We would like the network's state of activation to represent both the animal Tweety and the animal Sylvester. This cannot happen if concepts are represented by a single ρ-btu node because the node *animal* cannot fire in synchrony with both *Tweety* and *Sylvester* at the same time. Second, the encoding must provide *built-in* mechanisms for *automatically* controlling the direction of activation in the *IS-A* hierarchy so as to deal correctly with queries containing existentially and universally quantified variables. The correct treatment of quantified variables – assuming that all *IS-A* links are indefeasible, that is, without exceptions[22] – requires that activation originating from a concept *C* that is either an entity or the type corresponding to a universally quantified variable in the query should propagate upwards to all the ancestors of *C*. The upward propagation checks if the relevant fact is universally true of some superconcept of *C*. The activation origination from a concept *C* that appears as an existentially quantified variable in the query should propagate to the ancestors of *C*, the descendants of *C*, as well as the ancestors of the descendants of *C*.[23] A possible solution to these problems has been proposed by Mani and Shastri (1991) and is outlined below.

5.2. Encoding of the IS-A hierarchy

Each concept *C* represented in the *IS-A* hierarchy is encoded by a group of nodes called the *concept cluster* for *C* (see middle of Fig. 21). The concept cluster for *C* has k_1 *banks* of ρ-btu nodes, where k_1 is the *multiple instantiation constant* and refers to the number of dynamic instantiations a concept can accommodate. In general, the value

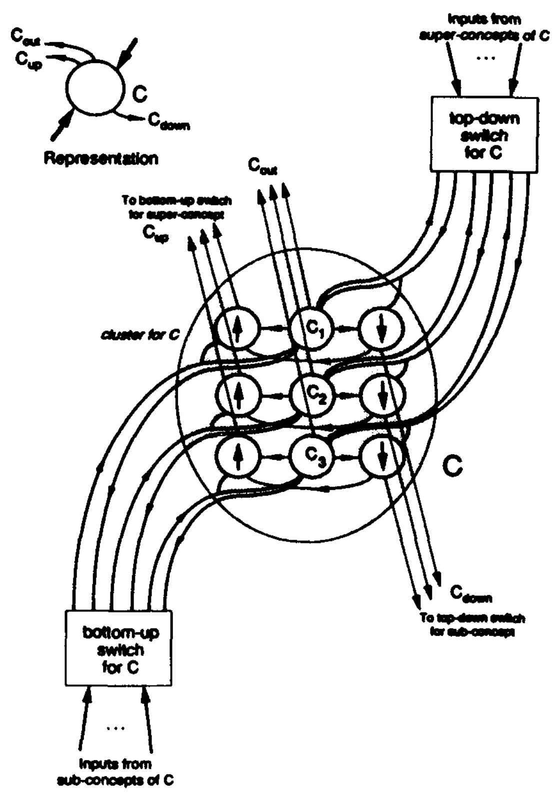


Figure 21. Structure of the concept cluster for C and its interaction with the bottom-up and top-down switches. The cluster has three banks of nodes and is capable of storing up to three distinct instances of the concept (in other words, the multiple instantiation constant k_1 equals three). The $\uparrow$ and $\downarrow$ relay nodes have a threshold of 2.

of k_1 may vary from concept to concept, but for ease of exposition we will assume that it is the same for all concepts. In Figure 21, k_1 is three. Each ***bank*** of concept C consists of three ρ-btu nodes: C_i, $C_{i\uparrow}$, $C_{i\downarrow}$. Each C_i can represent a distinct (dynamic) instantiation of C. The ***relay*** nodes $C_{i\uparrow}$ and $C_{i\downarrow}$ control the direction of the propagation of activation from C_i. The nodes $C_{i\uparrow}$ and $C_{i\downarrow}$ have a threshold of two. Note that C_i is connected to $C_{i\uparrow}$ and $C_{i\downarrow}$, and $C_{i\downarrow}$ is linked to $C_{i\uparrow}$.

Every concept C is associated with two subnetworks – the ***top-down*** and ***bottom-up*** switches. These switches are identical in structure and automatically control the flow of activation to the concept cluster. A switch has k_1 outputs. ***Output**$_i$ $(1 \le i \le k_1)$* from the bottom-up switch connects to C_i and $C_{i\uparrow}$, whereas ***output**$_i$* from the top-down switch goes to nodes C_i and $C_{i\downarrow}$. the bottom-up switch has $k_1 n_{sub}$ inputs and the top-down switch has $k_1 n_{sup}$ inputs, where n_{sub} and n_{sup} are the number of sub- and superconcepts of C, respectively. There are also links from the C_i nodes to both switches. The interaction between the switches and the concept cluster brings about efficient and automatic dynamic allocation of banks in the concept cluster by ensuring that (1) activation gets channeled to the concept cluster banks only if any "free" banks are available, and (2) each instantiation occupies only one bank.

The architecture of the switch (with k_1 = 3) is illustrated in Figure 22. The k_1 ρ-btu nodes, $S_1, \ldots, S_{k_1}$, with their associated τ-or nodes form the switch. Inputs to the switch make two connections – one excitatory and one inhibitory – to each of $S_2, \ldots, S_{k_1}$. As a result of these excitatory-inhibitory connections, nodes $S_2, \ldots, S_{k_1}$ are initially disabled and cannot respond to incoming activation. Any input activation only affects node S_1, because the switch inputs directly connect to S_1. S_1 becomes active in response to the first available input and continues to fire in phase with the input as long as the input persists. As S_1 becomes active, the τ-or node associated with S_1 turns on and enables S_2. However, inhibitory feedback from C_1 ensures that S_2 is *not* enabled in the phase in which C_1 is firing. Thus S_2 can start firing only in a different phase. Once S_2 starts firing, S_3 gets enabled, and so on.

Note that C_i could receive input in two phases, one from its bottom-up switch and another from its top-down switch. C_i, being a ρ-btu node, fires in only one of these phases. At any stage, if C_i, $1 \le i \le k_1$ picks up activation channeled by the other switch, feedback from C_i into the τ-or node associated with S_i causes S_{i+1} to become enabled, even though S_i may not be firing. The net result is that as instantiations occur in the concept cluster, the ρ-btu nodes in the switch get enabled, in turn, from left to right in distinct phases.

An *IS-A* relation of the form *is-a(A, B)* is represented as shown in Figure 23 by (1) connecting the $A_{i\uparrow}$, $i = 1, \ldots, k_1$ nodes to the bottom-up switch for B, and (2) connecting the $B_{i\downarrow}$, $i = 1, \ldots, k_1$ nodes to the top-down switch for A.

Consider a concept C in the *IS-A* hierarchy. Suppose C_i receives activation from the bottom-up switch in phase ρ. In response, C_i starts firing in synchrony with this activation. The $C_{i\uparrow}$ node now receives two inputs in phase ρ (one from the bottom-up switch and another from C_i; see Fig. 21). Since it has a threshold of 2, $C_{i\uparrow}$ also starts firing in phase ρ. This causes activation in phase ρ to eventually spread to the superconcept of C. Hence any upward-traveling activation continues to travel upward, which is the required behavior when C is associated with a universally typed variable. Similarly, when C_i receives activation from the top-down switch in phase ρ, both C_i and $C_{i\downarrow}$ become active in phase ρ. $C_{i\uparrow}$ soon follows suit because of the link from $C_{i\downarrow}$ to $C_{i\uparrow}$. Thus eventually the whole i^{th} bank starts firing in phase ρ. This built-in mechanism allows a concept associated with an existentially typed variable to eventually

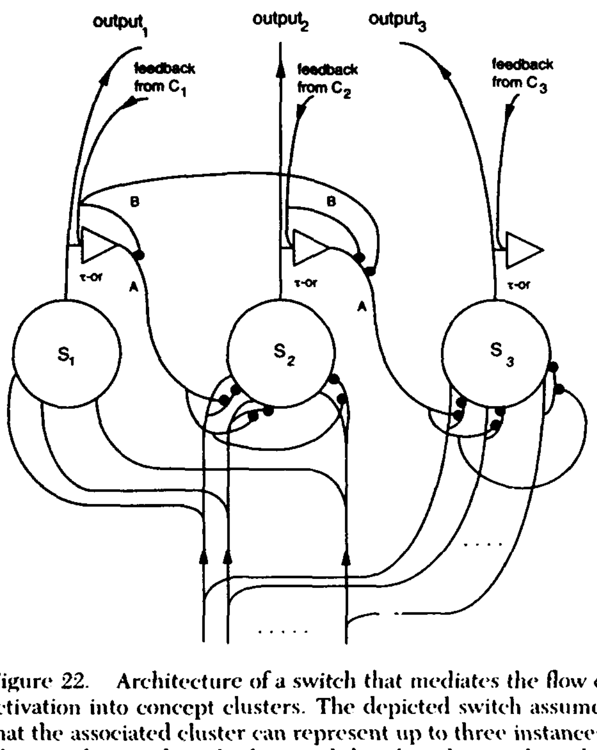


Figure 22. Architecture of a switch that mediates the flow of activation into concept clusters. The depicted switch assumes that the associated cluster can represent up to three instances. The switch provides a built-in and distributed control mechanism for automatically allocating banks within a concept cluster. Each distinct incoming instantiation is directed to a distinct bank, provided one is available.

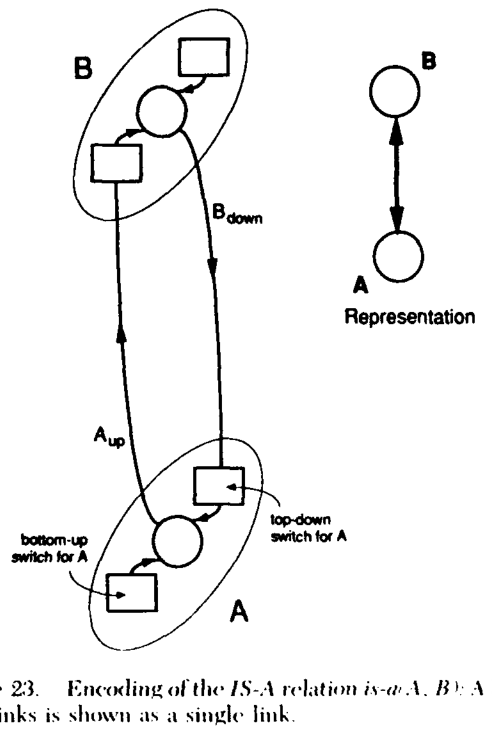


Figure 23. Encoding of the *IS-A* relation *is-a(A, B)*: A bundle of k_1 links is shown as a single link.

spread its activation to its ancestors, descendants, and ancestors of descendants. The switching mechanism introduces an extra delay in the propagation of activation along *IS-A* links and, typically, the switch takes three steps to channel the activation. In the worst – and also the least likely – case, the switch may take up to eight steps to propagate activation.

The time taken to perform inferences in the integrated system is also independent of the size of the LTKB and is proportional only to the length of the shortest derivation of the query. The time taken to perform a predictive inference is approximately $l_1\pi + 3l_2\pi$, where l_1 and l_2 are the lengths of the shortest chain of rules, and the shortest chain of *IS-A* links, respectively, that must be traversed in order to perform the inference. The time required to answer a yes-no query is approximately $max(l_1\pi, 3l_2\pi) + l_1\pi + 2\pi$.[24]

5.3. Typed variables in queries

Consider a query $P(\ldots, x, \ldots)?$, where the j^{th} argument is specified as a typed variable x. If x is universally quantified, that is, the query is of the form $\forall x{:}\ C\ P(\ldots, x, \ldots)$, C_i and $C_{i\uparrow}$ are activated in phase with the j^{th} argument node of P (the subscript i refers to one of the banks of C). If x is existentially quantified, that is the query is of the form $\exists x : C\ P(\ldots, x, \ldots)$, C_i and $C_{i\downarrow}$ are activated in phase with the j^{th} argument node of P. As before (sect. 4.3), an untyped variable in a query is not activated. Simple queries of the type *is-a(C, D)?* are posed by simply activating the nodes C_i and $C_{i\uparrow}$ and observing whether one or more D_is become active.

5.4. Encoding appropriateness as type restrictions on argument fillers

The *IS-A* hierarchy can be used to impose type restrictions on variables occurring in rules. This allows the system to encode context-dependent rules that are sensitive to the types of the argument fillers involved in particular situations. Figure 24 shows the encoding of the following rule in a forward-reasoning system: $\forall x$*:animate, y:solid-obj walk-into(x,y)* $\Rightarrow$ *hurt(x)* (i.e., If an animate agent walks into a solid object, the agent gets hurt). The types associated with variables specify the admissible types (categories) of fillers, and the rule is expected to fire only if the fillers bound to the arguments are of the appropriate type. The encoding makes use of τ-or nodes that automatically check whether the filler of an argument is of the appropriate type. Thus the τ-or node *a* in Figure 24 would become active if and only if the first argument of *walk-into* is firing in synchrony with *animate*, indicating that the filler of the argument is of type *animate*. Similarly, the τ-or node *b* would become active if and only if the second argument of *walk-into* is firing in synchrony with *solid-object*, indicating that the filler of this argument is of type *solid-object*. The activation of nodes *a* and *b* would enable the propagation of activity from the antecedent to the consequent predicate. In a forward reasoner, typed variables are allowed only in the antecedent of the rule.

In the backward reasoner, typed variables are allowed only in the consequent of a rule. The encoding of a typed universally quantified variable in the consequent is similar to the encoding of an entity in the consequent of a rule explained in section 4.5 (see Fig. 14). Instead of originating at an entity, the inhibitory link originates at the concept representing the type of the universally quantified variable. The encoding of a typed existentially quantified variable is similar to that of a typed universally quantified variable except that the inhibitory link originates from a unique subconcept of the associated concept (for details refer to Mani & Shastri 1991).

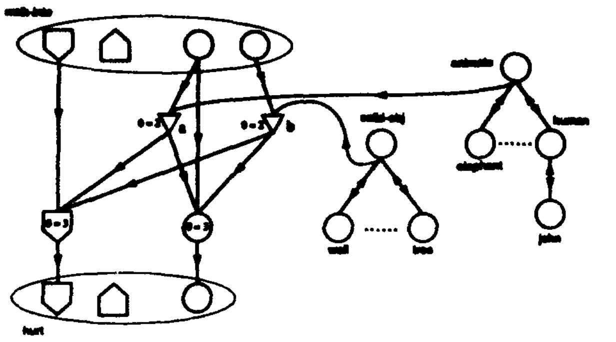


Figure 24. Encoding rules with typed variables: The network fragment encodes the rule $\forall x$ *: animate,y : solid-obj walk-into(x, y)* $\Rightarrow$ *hurt(x)*. The numbers associated with nodes denote thresholds (only thresholds other than 1 have been indicated explicitly). The τ-or node *a (b)* become active if and only if the first (second) argument node of *walk-into* fires in synchrony with the concept *animate (solid-obj)*. Once active, these nodes enable the propagation of binding to the predicate *hurt*. Thus type restrictions are enforced using temporal synchrony.

The rule $\forall$ *x:animate, y:solid-obj walk-into(x,y)* $\Rightarrow$ *hurt(x)* is logically equivalent to the rule $\forall$ *x,y animate(x)* $\wedge$ *solid-obj(y)* $\wedge$ *walk-into(x,y)* $\Rightarrow$ *hurt(x)*. Thus it would appear that the *IS-A* hierarchy is not essential for encoding type restrictions on rules. Note, however, that although the former variant has only one predicate in the antecedent, the latter has three. This increase in the number of antecedent predicates can be very costly, especially in a forward-reasoning system capable of supporting multiple dynamic predicate instantiations (Mani & Shastri 1992). In such a system, the number of nodes required to encode a rule is proportional to k_2^m, where k_2 is the bound on the number of times a predicate may be instantiated dynamically during reasoning (see sect. 6), and m equals the number of predicates in the antecedent of the rule. Thus it is critical that m be very small. The *IS-A* hierarchy plays a crucial role in reducing the value of m by allowing restrictions on predicate arguments to be expressed as type restrictions.[25]

5.5. Encoding soft and defeasible rules

The proposed solution to the binding problem can be generalized to soft/defeasible rules. Observe that the strength of dynamic bindings may be represented by the degree of synchronization between an argument and a filler (this possibility was suggested by Jed Harris, personal communication). Such a scheme becomes plausible if each argument is encoded by an ensemble of nodes (see sect. 7.3), for then the degree of coherence in the phase of firing of nodes within an argument ensemble can indicate the strength of the binding the argument is participating in. In the limiting case, a highly dispersed activity in an argument ensemble may mean that the argument is bound to one of the active entities, although it is not clear which (Shastri 1993a).[26]

In addition to specifying a mechanism for representing the strength of dynamic bindings and rule firing, we also need to specify the basis for computing these strengths. It is customary to view the strength of a rule as a numerical quantity associated with a rule (e.g., *certainty factors* in MYCIN; Buchanan & Shortliffe 1984). Such as "atomic" and uninterpreted view of the strength of a rule is inadequate for modeling rules involving *n*-ary predicates. Our approach involves defining the "strength" of a rule (and similarly, the strength of a binding) to be a dynamic quantity that depends upon the types/features of the entities bound to arguments in the rule at the time of rule application. Such a strength measure also generalizes the notion of type restrictions on rules to type preferences on rules. Thus instead of rules of the form $\forall x$: *animate*, *y* : *solid-obj walk-into(x,y)* $\Rightarrow$ *hurt(x)*, the system can encode rules of the form:

$$\forall x,y\ \textit{walk-into}(x,y) \Rightarrow [\text{with strength } \sigma(\textit{type}(x),\textit{type}(y))] \Rightarrow \textit{hurt}(x),$$

where the strength of the rule may vary from one situation to another as a function, σ, of the types of the argument fillers in a given situation. Observe that the value of $\sigma(t_i, t_j)$ need not be known for all types t_i and t_j in the *IS-A* hierarchy, and may be inherited. For example, if $\sigma(t_1, t_2)$ is not known but $\sigma(t_m, t_n)$ is, and t_1 and t_2 are subtypes of t_m and t_n, respectively, then $\sigma(t_m, t_n)$ can be used in place of $\sigma(t_1, t_2)$. This is analogous to property inheritance in an *IS-A* hierarchy, where property values may be attached to just a few concepts and the property values of the rest of the concepts inferred via inheritance. The proposed treatment would allow the system to incorporate exceptional and default information during reasoning. This relates to Shastri's (1988a; 1988b) work on a connectionist semantic network (see sect. 9.2).

6. Representing multiple dynamic instantiations of a predicate

The representation of dynamic bindings described thus far cannot simultaneously represent multiple dynamic facts about the same predicate. The proposed representation can be extended to do so by generalizing the scheme for encoding multiple instantiations of concepts outlined in section 5.2. The extension assumes that during an episode of reflexive reasoning each predicate can be instantiated only a bounded number of times. In general, this bound may vary across predicates and some critical predicates may have a marginally higher bound. For ease of exposition, however, we will assume that this bound is the same for all predicates and refer to it as k_2. The ability to handle multiple instantiations of the same predicate allows the system to deal with more complex inferential dependencies, including circularities and bounded recursion. The system can make use of rules such as $\forall x,y$ *sibling* $(x,y) \Rightarrow$ *sibling* (y,x). A forward-reasoning system can use a rule such as $\forall x,y,z$ *greater* $(x,y) \wedge$ *greater*(y,z) $\Rightarrow$ *greater*(x,z) and infer "*a* is greater than *c*" on being told "*a* is greater than *b*" and "*b* is greater than *c*."

Since up to k_2 dynamic instantiations of a predicate may have to be represented simultaneously, the representation of an *n*-ary predicate is augmented so that each predicate is represented by k_2 *banks* of nodes, with each bank containing a *collector*, an *enabler*, and *n* argument nodes. For a given predicate, *P*, the enabler of the i^{th} bank $e{:}P_i$ will be active whenever the i^{th} bank has been instantiated with some dynamic binding. The collector $c{:}P_i$ of the i^{th} bank will be activated whenever the dynamic bindings in the i^{th} bank match the knowledge encoded in the LTKB. Figure 25 depicts the encoding of two binary predicates *P* and *Q* and a ternary predicate *R*.

Given that a predicate is represented using multiple banks of predicate and argument nodes, the connections between arguments of the antecedent and consequent predicates of a rule have to be mediated by a "switching" mechanism similar to the one described in section 5.2. The switch automatically channels input instantiations into available banks of its associated predicate. It also ensures that each distinct instantiation occupies only one bank, irrespective of the number of consequent predicates that may be communicating this instantiation to the switch.

With the inclusion of the switch in the backward reasoning system, the number of nodes required to represent a predicate and a long-term fact becomes propor-

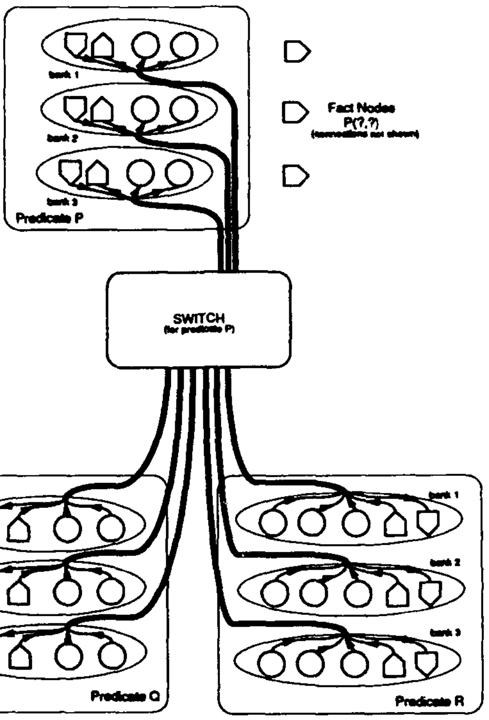


Figure 25. The encoding of predicates for accommodating multiple instantiations: *P* and *Q* are binary predicates and *R* is a ternary predicate. The encoding assumes that any predicate may be instantiated at most three times (i.e., the multiple instantiation constant $k_2 = 3$). An *n*-ary predicate is represented by k_2 banks of nodes. The connections suggest that there are two rules, one of the form $P() \Rightarrow Q()$ and the other of the form $P() \Rightarrow R()$ (the argument correspondence is not shown). The connections between antecedent and consequent predicates of a rule are mediated by a "switching" mechanism similar to the one described in Figure 22. The switch for *P* automatically channels incoming instantiations of *P* into available banks of *P*. The switch has k_2 output "cables," where each cable consists of output links to a predicate bank of *P*. The inputs to the switch are cables from banks of predicates that are in the consequent of rules in which *P* occurs in the antecedent.

tional to k_2 and the number of nodes required to encode a rule becomes proportional to k_2^2. Furthermore, the time required for propagating multiple instantiations of a predicate increases by a factor of k_2. Thus there are significant space and time costs associated with multiple instantiation of predicates. The complete realization of the switch and its interconnection is described in Mani and Shastri (1992).

7. Biological plausibility

Recent neurophysiological data suggest that synchronous, rhythmic activity occurs in the brain and that the time course of such activity is consistent with the requirements of reflexive reasoning. The data also provide evidence in support of the hypothesis that the cat visual system solves the dynamic-binding problem by using temporal synchrony.

7.1. Neurophysiological support

There is considerable evidence for the existence of rhythmic activity in the animal brain. Synchronous activity has been documented for some time in the olfactory bulb, hippocampus, and the visual cortex (Freeman 1981; Gerstein 1970; MacVicar & Dudek 1980; Toyama et al. 1981). The most compelling evidence for such activity, however, comes from findings of synchronous oscillations in the visual cortex of anesthetized cats responding to moving visual stimuli (Eckhorn et al. 1988; 1990; Engel et al. 1991; Gray & Singer 1989; Gray et al. 1989; 1991). These findings are based on the correlational analysis of local field potentials and multiunit, as well as single-unit, recordings. Recently, similar activity has also been reported in an awake and behaving monkey (Kreiter & Singer 1992).[27] Relevant aspects of the experimental findings are summarized below:

1. Synchronous oscillations have been observed at frequencies ranging from 30 to 80 Hz (a typical frequency is around 50 Hz).
2. Synchronization of neural activity can occur within a few periods (sometimes even one period) of oscillations (Gray et al. 1991).
3. In most cases synchronization occurs with a lag or lead of less than 3 msec, although in some cases it even occurs with precise phase locking (Gray et al. 1991).
4. Synchronization of neural activity occurs (a) between local cortical cells (Eckhorn et al. 1988; Gray & Singer 1989), (b) among distant cells in the same cortical area (Gray et al. 1989), (c) among cells in two different cortical areas – for example, areas 17 and 18 (Eckhorn et al. 1988) and areas 17 and PMLS (Engel et al. 1991), and (d) among cells across the two hemispheres (Engel et al. 1991).
5. Once achieved, synchrony may last several hundred msec (Gray et al. 1991).

The synchronous activity observed in the brain is a complex and dynamic phenomenon. The frequency and degree of phase locking varies considerably over time and the synchronization is most robust when viewed as a property of groups of neurons. The nature of synchronous activity assumed by our model is an idealization of such a complex phenomenon (but see sects. 7.3 & 10.1–10.4).

7.1.1. Temporal synchrony and dynamic bindings in the cat visual cortex.

Neurophysiological findings also support the hypothesis that the dynamic binding of visual features pertaining to a single object may be realized by the synchronous activity of cells encoding these features (see Eckhorn et al. 1990; Engel et al. 1991). In one experiment, multiunit responses were recorded from four different sites that had overlapping receptive fields but different orientation preferences – 157^0, 67^0, 22^0, and 90^0, respectively. A vertical light bar resulted in a synchronized response at sites 1 and 3, whereas a light bar oriented at 67^0 led to a synchronized response at sites 2 and 4. A combined stimulus with the two light bars superimposed led to activity at all the four sites, but although the activity at sites 1 and 3 was synchronized and that at sites 2 and 4 was synchronized, there was no correlation in the activity across these pairs of sites (Engel et al. 1991). Such experimental evidence suggests that the synchronous activity in orientation specific cells may be the brain's way of encoding that these cells are participating in the representation of a single object. This is analogous to the situation in Figure 9, wherein the synchronous activity of the nodes *recip*, *owner*, and *cs-seller* in phase with *Mary* is the system's way of encoding that all these roles are being filled by the same object, Mary.

7.2. Some neurally plausible values of system parameters

The neurophysiological data cited above also provide a basis for making coarse but neurally plausible estimates of some system parameters. The data indicate that plausible estimates of π_{min} and π_{max} may be 12 and 33 msec, respectively, and a typical value of π may be 20 msec. The degree of synchronization varies from episode to episode, but a conservative estimate of ω, the width of the window of synchrony, may be derived on the basis of the cumulative histogram of the phase difference (lead or lag) observed in a large number of synchronous events. The standard deviation of the phase differences was 2.6 msec in one data set and 3 msec in another (Gray et al. 1991). Thus a plausible estimate of ω may be about 6 msec. Given that the activity of nodes can get synchronized within a few cycles, sometimes even within one, and given the estimates of π_{min} and π_{max}, it is plausible that synchronous activity can propagate from one ρ-btu node to another in about 50 msec. The data also suggest that synchronous activity lasts long enough to support episodes of reflexive reasoning requiring several steps.

7.3. Propagation of synchronous activity – a provisional model

Our system requires the propagation of synchronous activity over interconnected nodes in spite of nonzero and noisy propagation delays. The neurophysiological evidence cited in the previous sections suggest that such propagation occurs in the cortex. The exact neural mechanisms underlying the propagation of such activity, however, remain to be determined. Abeles (1982; 1991) has argued, on the basis of anatomical and physiological data, theoretical analysis, and simulation results that synchronous activity can propagate over chains of neurons connected in a many-to-many fashion (synfire chains) with a small and stable "jitter," even if random fluctuations are taken into account. Bienenstock (1991) has examined how synfire chains may arise in networks as a result of learning. Below we outline a provisional model (Mandelbaum & Shastri 1990) that demonstrates that synchronized activity can propagate in spite of noisy propagation delays. The model is meant to demonstrate the feasibility of such propagation and should not be viewed as a detailed neural model.

We assume that each argument in the reasoning system is represented by an ensemble of n nodes rather than just a single node. Connections between arguments are encoded by connecting nodes in the appropriate ensembles: If ensemble A connects to ensemble B, then each node in A is randomly connected to m nodes in B ($m \leq n$). Thus, on average, each node in B receives inputs from m nodes in A (see Fig. 26) and has a threshold comparable to m. The propagation delay between nodes in two different ensembles is assumed to be noisy and is modeled as a Gaussian distribution. If ensemble A is connected to ensemble B and nodes in ensemble A are firing in synchrony, then we desire that within a few periods of oscillation nodes in ensemble B start firing in synchrony with nodes in ensemble A.

Nodes within an ensemble are also sparsely interconnected, with each node receiving inputs from a few neighboring nodes within the ensemble. Synchronization within an ensemble is realized as a result of the interaction between the feedback received by a node from its neighbors within the ensemble. The model makes use of the negative slope of the threshold-time characteristic during the *relative refractory period* (RRP) to modulate the timing of the spike generated by a node. Observe that a higher excitation can hasten, while a lower excitation can delay, the firing of a node. At the same time, the lag between the firing times of nodes in two connected ensembles due to the mean propagation delay is partially offset by the interaction between the various parameters of the system enumerated below.

The threshold-time characteristic of a node and the distribution of the arrival times of input spikes from a preceding ensemble are illustrated in Figure 27. After firing, a node enters an absolute refractory period (ARP), in which its threshold is essentially infinite. The ARP is followed by a decaying RRP, during which the threshold decays to its resting value. During the RRP, the threshold-time characteristic is approximated as a straight line of gradient g (a linear approximation is not critical). The incoming spikes from a preceding ensemble arrive during a node's ARP and the early part of its RRP. It is assumed that immediate neighbors within an ensemble can rapidly communicate their potential to each other.

A node's potential is the combined result of the interensemble and intraensemble interactions and in the period between spikes is modeled as

$$P_i(t + \Delta t) = P_i(t) + In_i(t) + \alpha * \Sigma_j[P_i(t) - P_j(t)],$$

where $P_i(t)$ is the potential of node i at time t. The change in potential is caused by $In_i(t)$, the input arriving at node i from nodes in the preceding ensemble as well as the difference in the potential of i and that of its immediate neighbors j. In the simulation, j ranged over *six* immediate neighbors of i. If nodes i and j are immediate neighbors and i is firing ahead of j, then we want i to hasten the firing of j by sending it an excitatory signal and j to delay the firing i by sending it an inhibitory signal. Doing so would raise the potential of j, causing it to fire early, and lower the potential of i, causing it to fire later in the next cycle. Thus i and j would tend to synchronize.[28]

The results of a sample simulation are shown in Figure 28. The diagram shows the cycle-by-cycle distribution of

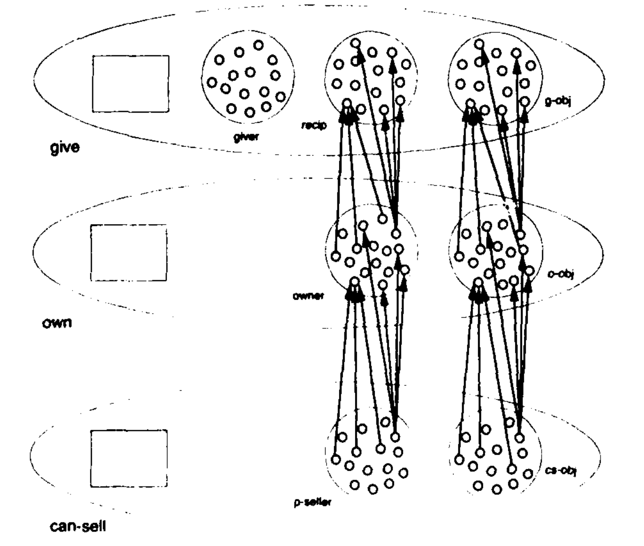


Figure 26. Individual ρ-btu nodes are replaced by an ensemble of such nodes. A connection between a pair of individual ρ-btu nodes is replaced by a number of random interensemble connections. Nodes within an ensemble can communicate with their immediate neighbors in the ensemble and the intraensemble propagation delays are assumed to be much smaller than the interensemble propagation delays.

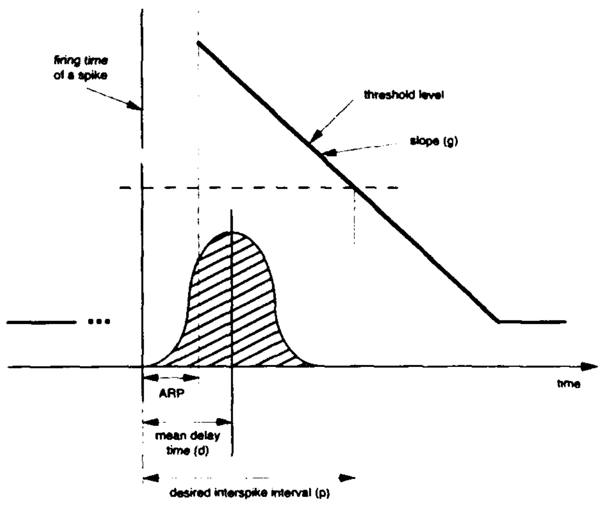


Figure 27. The time course of a node's threshold. After generating a spike a node enters an absolute refractory period (ARP). The ARP is followed by a relative refractory period (RRP). After the RRP a node's threshold reverts to its normal level. The distribution of the arrival times of signals from a connected ensemble is depicted by the shaded region. The noisy propagation delays are modeled as a Gaussian.

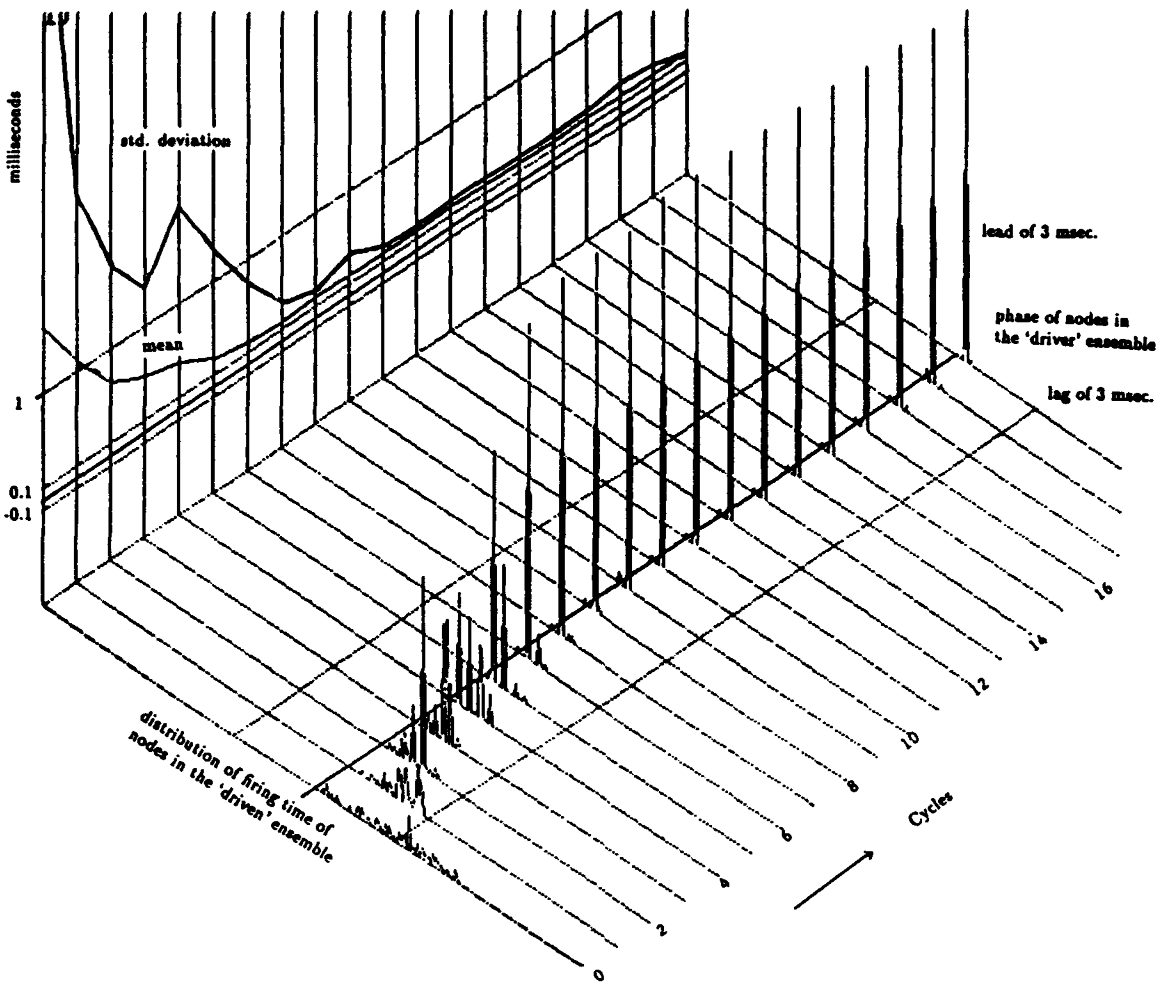


Figure 28. The cycle-by-cycle distribution of the firing times of nodes within a "driven" ensemble being driven by a "driver" ensemble whose nodes are firing in synchrony. The left-hand "wall" of the isometric diagram displays the standard deviation and mean of the node firing times with reference to the ideal firing time. The nodes in the driven ensemble become synchronized in spite of noisy propagation delays. The *maximum* lag in the firing times of nodes in the driven ensemble becomes less than 3 msec and the *mean* lag becomes less than 1 msec within two cycles. By the end of seven cycles the maximum and mean lags reduce to 1 and 0.2 msec, respectively.

the firing times of nodes within a "driven" ensemble that is being driven by a "driver" ensemble whose nodes are oscillating in phase lock. Δt was chosen to be 0.001 time units (*i.e.*, all calculations were done every 1/1000 of a time unit), where a unit of time may be assumed to be 1 msec. Other simulation parameters were as follows: (1) n, the number of nodes in ensemble, equals 64; (2) m, the interensemble connectivity, equals 20; (3) g, the slope of the threshold during the RRP, equals 0.032; (4) α, the "coupling" factor between immediate neighbors within an ensemble, equals 0.07; (5) d, the average interensemble propagation delay, equals 4.5 time units; (6) s, the standard deviation of interensemble propagation delay, equals 1.5 time units; and (7) π, the expected period of oscillation, equals 10.5 time units.

As shown in Figure 28, despite noisy propagation delays, the *maximum* lag in the firing of nodes in the "driven" ensemble becomes less than 3 msec and the *mean* lag becomes less than 1 msec within two cycles. By the end of seven cycles the maximum and mean lags reduce to 1 and 0.2 msec, respectively.

8. Psychological implications

In this section we examine the psychological implications of our system, especially in view of the biologically motivated estimates of the system parameters discussed in section 7.3.

8.1. A neurally plausible model of reflexive reasoning

The proposed system can encode specific as well as general instantiation-independent knowledge and perform a broad range of reasoning with efficiency. The system makes use of very simple nodes, and yet its node requirement is only linear in the size of the LTKB (the size being measured in terms of the number of predicates, facts, rules, concepts, and *IS-A* relations). Thus the system illustrates how a large LTKB may be encoded by using only a fraction of 10^{12} nodes.

The system demonstrates that a class of forward and backward reasoning can be performed very rapidly, in time independent of the size of the LTKB. Below we set the values of appropriate system parameters to neurally plausible values identified in section 7.3 and indicate the time the system takes to perform certain inferences.[29] These times are, at best, broad indicators of the time we expect the internal reasoning process to take. Also note that they do not include the time that would be taken by perceptual, linguistic, and motor processes to process and respond to inputs.

8.1.1. Some typical retrieval and inference timings. Let us choose π to be 20 msec and assume that ρ-btu nodes can synchronize within two periods of oscillations. The system takes 320 msec to infer "John is jealous of Tom" after being given the dynamic facts "John loves Susan" and "Susan loves Tom" (this assumes the rule, if x loves y and y loves z then x is jealous of z). The system takes 260 msec to infer "John is a sibling of Jack," given "Jack is a sibling of John." Similarly, the system takes 320 msec to infer "Susan *owns* a car" after its internal state is initialized to represent "Susan *bought* a Rolls-Royce." If the system's LTKB includes the long-term fact "John bought a Rolls-Royce," the system takes 140, 420, and 740 msec, respectively, to answer yes to the queries Did John buy a Rolls-Royce? Does John own a car? and Can John sell a car?

Thus our system demonstrates that a class of reasoning can occur rapidly, both in the forward (predictive) mode as well as backward (query answering) mode. The above times are independent of the size of the LTKB and do not increase when additional rules, facts, and *IS-A* relationships are added to the LTKB. If anything, these times may decrease if one of the additional rules is a composite rule and short-circuits an existing inferential path. For example, if a new rule "if x buys y then x can sell y" were to be added to the LTKB, the system would answer the query Can John sell a car? in 420 msec instead of 740 msec.

8.1.2. Variations in inference and retrieval times. Consider two ρ-btu nodes A and B such that A is connected to B (although we are referring to individual nodes, the following comment would also apply if A and B were ensembles of nodes). It seems reasonable to assume that the number of cycles required for B to synchronize with A will depend on the synaptic efficacy of the link from A to B. This suggests that the time taken by the propagation of bindings – and hence rule firing – will vary, depending on the weights on the links between the argument nodes of the antecedent and consequent predicates. Rules whose associated links have high weights will fire and propagate bindings faster than rules whose associated links have lower weights. It also follows that different facts will take different times to be retrieved, depending on the weights of the links connecting the appropriate arguments and filler concepts (see Fig. 10). Note that the inhibitory signal from an argument will continue to block the activation of a fact node until the signals from the filler concepts and the argument get synchronized. Similarly, during the processing of *wh*-queries, the time it would take for the filler concepts to synchronize with the binder units will depend on the weights of the links from the binder nodes to the concept nodes (see Fig. 17). Thus the retrieval of argument fillers would be faster if the weights on the appropriate links are high.[30] Observe that the variation in times refers to the variations in the time it takes for nodes to *synchronize* and not the time it takes for nodes to become active. This suggests that the time course of systematic inferences and associative priming may be quite different.

8.2. Nature of reflexive reasoning

Our model suggests several constraints on the nature of reflexive reasoning. These have to do with (1) the capacity of the working memory underlying reflexive reasoning, (2) the form of rules that may participate in such reasoning, and (3) the depth of the chain of reasoning.

8.2.1. The working memory underlying reflexive reasoning. Dynamic bindings and, hence, dynamic facts are represented in the system as a rhythmic pattern of activity over nodes in the LTKB network. In functional terms, this transient state of activation can be viewed as a limited-capacity dynamic *working memory* that temporarily holds information during an episode of reflexive reasoning. Let us refer to this working memory as the WMRR.

Our system predicts that the capacity of the WMRR is very large and, at the same time, very limited! The number of dynamic facts that can simultaneously be present in the WMRR is high and is given by k_2p, where k_2 is the predicate multiple instantiation constant introduced in section 6, and p is the number of predicates represented in the system. The number of concepts that may be active simultaneously is also very high and equals k_1c, where c is the number of concepts in the *IS-A* hierarchy and k_1 is the multiple instantiation constant for concepts introduced in section 5.2. But, as we discuss below, there are two constraints that limit the number of dynamic facts that may actually be present in the WMRR at any given time.

8.2.2. Working memory, medium-term memory and overt short-term memory. Before moving on let us make two observations. First, the dynamic facts represented in the WMRR during an episode of reasoning should not be confused with the small number of short-term facts that an agent may *overtly* keep track of during reflective processing and problem solving. In particular, the WMRR should not be confused with the (overt) short-term memory implicated in various memory span tasks (for review see Baddeley 1986). Second, our reasoning system implies that a large number of dynamic facts will be produced as intermediate results during reasoning and would be represented in the WMRR. These facts, however, are only potentially relevant and would remain covert and decay in a short time unless they turn out to be relevant in answering a "query" or providing an explanation. We expect that only a small number of dynamic facts would turn out to be relevant, and those that do would "enter" a medium-term memory, where they would be available for a much longer time (see sect. 10.5). Some of these facts may also enter the overt short-term memory. Note that this short-term memory need not be a physically distinct module. It may simply consist of facts/entities in the WMRR that are currently receiving an attentional spotlight (cf. Crick 1984; Crick & Koch 1990a).

8.2.3. A bound on the number of distinct entities referenced in the WMRR. During an episode of reasoning, each entity involved in dynamic bindings occupies a distinct phase in the rhythmic pattern of activity. Hence the number of distinct entities that can occur as argument fillers in the dynamic facts represented in the WMRR cannot exceed $\lfloor \pi_{max}/\omega \rfloor$, where π_{max} is the maximum period (corresponding to the lowest frequency) at which ρ-btu nodes can sustain synchronous oscillations, and ω equals the width of the window of synchrony. Thus the WMRR may represent a large number of facts, as long as these facts refer to only a small number of distinct entities. Note that the activation of an entity together with all its active superconcepts counts as only one entity.

In section 7.2 we pointed out that a neurally plausible value of π_{max} is about 33 msec and a conservative estimate of ω is around 6 msec. This suggests that as long as the number of entities referenced by the dynamic facts in the WMRR is five or less, there will essentially be no cross-talk among the dynamic facts. If more entities occur as argument fillers in dynamic facts, the window of synchrony ω would have to shrink in order to accommodate all the entities. For example, ω would have to shrink to about 5 msec in order to accommodate 7 entities. As ω shrinks, the possibility of cross-talk between dynamic bindings would increase and eventually disrupt the reasoning process. The exact bound on the number of distinct entities that may fill arguments in dynamic facts would depend on the smallest feasible value of ω. Given the noise and variation indicated by the data on synchronous activity cited in section 7.1, it appears unlikely that ω can be less than 3 msec. Hence we predict that a neurally plausible upper bound on the number of distinct entities that can be referenced by the dynamic facts represented in the WMRR is about 10. This prediction is consistent with our belief that most cognitive tasks performed without deliberate thought tend to involve only a small number of distinct entities at a time (though, of course, these entities may occur in multiple situations and relationships).

It is remarkable that the bound on the number of entities that may be referenced by the dynamic facts in the WMRR relates so well to 7 ± 2, the robust measure of short-term memory capacity (Miller 1956). This unexpected coincidence merits further investigation as it suggests that temporal synchrony may also underlie other short-term and dynamic representations. Similar limitations of the human dynamic-binding mechanism are also illustrated in experimental work on the attribute-binding problem (Stenning et al. 1988).

The bound on the number of distinct entities referenced in the WMRR is independent of similar bounds on the working memories of other subsystems. As we discuss in section 10.4, dynamic structures in the working memory of other subsystems may refer to different sets of entities using phase distributions local to those subsystems.

8.2.4. A bound on the multiple instantiation of predicates. The capacity of the WMRR is also limited by the constraint that it may only contain a small number of dynamic facts pertaining to each predicate. This constraint stems from the high cost of maintaining multiple instantiations of a predicate. As stated in section 6, in a backward-reasoning system, if k_2 denotes the bound on the number of times a predicate may be instantiated during an episode of reasoning, then the number of nodes required to represent a predicate and the associated long-term facts is proportional to k_2, and the number of nodes required to encode a rule is proportional to k_2^2. Thus a backward-reasoning system that can represent three dynamic instantiations of each predicate will have anywhere from three to nine times as many nodes as a system that can only represent one instantiation per predicate. In a forward-reasoning system the cost is even higher and the number of nodes required to encode a rule is k_2^m, where m is the number of antecedents in the rule. The time required for propagating multiple instantiations of a predicate also increases by a factor of k_2. In view of the significant space and time costs associated with multiple instantiation and the necessity of keeping these resources within bounds in the context of reflexive reasoning, we predict that k_2 is quite small, perhaps no more than three. As observed in section 6, k_2 need not be the same for all predicates, and it is possible that some critical predicates may have a slightly higher k_2.[31]

8.2.5. Form of rules that may participate in reflexive reasoning. In section 4.9 we pointed out that when answering queries based on the long-term knowledge encoded in the LTKB, our reflexive-reasoning system cannot use rules that contain variables occurring in multiple argument positions in the *antecedent* unless such variables also appear in the consequent and get bound during the query-answering process. A similar constraint applies to forward (predictive) reasoning: When making predictions based on given dynamic facts, a system cannot use a rule that contains variables occurring in multiple argument positions in the *consequent*, unless such variables also appear in the antecedent and get bound during the reasoning process. These constraints predict that certain queries cannot be answered in a reflexive manner even though the corresponding predictions can be made reflexively. For example, consider an agent whose LTKB includes the rule "if x loves y and y loves z then x is jealous of z" and the long-term facts "John loves Mary" and "Mary loves Tom." Our system predicts that if this agent is asked Is John jealous of Tom? she will be unable to answer the query in a reflexive manner. Note that the antecedent of the rule includes a repeated variable, y, that does not occur in the consequent. Hence our system predicts that answering this question will require deliberate and conscious processing (unless the relevant long-term facts are active in the WMRR for some reason at the time the query is posed). However, an agent who has the above rule about love and jealousy in its LTKB would be able to infer "John is jealous of Tom" in a reflexive manner, on being "told" "John loves Mary" and "Mary loves Tom." This is because such an inference involves forward (predictive) reasoning.

As another example of the predictions made by the constraint, assume that our agent's conception of kinship relations is one wherein the maternal/paternal distinction at the grandparent level is not primary. Let us also assume that the agent's maternal grandfather is George. The constraint predicts that the agent cannot answer yes to the query Is George your maternal grandfather? in a reflexive manner even though the agent may be able to answer the question *Is George your grandfather?* in a reflexive manner (this example is due to Feldman, personal communication). The basis of this prediction is as follows: If "maternal grandfather" is not a primary kinship relation then it must be computed by using an appropriate rule. Given the nature of the maternal-grandfather relationship, any rule that does so would violate the repeated variable restriction.[32]

The restrictions imposed on the reasoning system also imply that it is not possible to apply the abstract notion of transitivity in a reflexive manner when answering queries. Observe that we need to state $\forall x,y,z\ P(x,y) \wedge P(y,z) \Rightarrow P(x,z)$ in order to assert that the relation P is transitive and the rule has the variable y occurring twice in the antecedent but not even once in the consequent. Given that transitivity plays an important role in commonsense reasoning – to wit, reasoning about sub- and supercategories, part-of relationships, greater and less than – the inability to handle transitivity might appear to be overly limiting. However, this is not the case. We believe that as far as query answering is concerned, humans are only good at dealing with the transitivity of a small number of relations. In these cases, the transitivity of the appropriate relations is encoded explicitly and the computation of transitivity does not require the use of an abstract transitivity rule. The organization of concepts in an *IS-A* hierarchy using *IS-A* links to capture the subclass/superclass relationship is an excellent case in point. The use of *IS-A* links converts the problem of computing the transitive closure from one of applying the transitivity rule $\forall x,y,z\ IS\text{-}A(x,y) \wedge IS\text{-}A(y,z) \Rightarrow IS\text{-}A(x,z)$, to one of propagating activation along links.

8.2.6. Bound on the depth of the chain of reasoning. Two things might happen as activity propagates along a chain of argument ensembles during an episode of reflexive reasoning. First, the lag in the firing times of successive ensembles may gradually build up due to the propagation delay introduced at each level in the chain. Second, the dispersion within each ensemble may gradually increase due to the variations in propagation delays and the noise inherent in synaptic and neuronal processes. Whereas the increased lag along successive ensembles will lead to a "phase shift" and, hence, binding confusions, the increased dispersion of activity within successive ensembles will lead to a gradual loss of binding information. Increased dispersion would mean less phase specificity and, hence, more uncertainty about the argument's filler. Because of the increase in dispersion along the chain of reasoning, the propagation of activity will correspond less and less to a propagation of argument bindings and more and more to an associative spread of activation. For example, the propagation of activity along the chain of rules $P_1(x,y,z) \Rightarrow P_2(x,y,z) \Rightarrow \ldots P_n(x,y,z)$ resulting from the input $P_1(a,b,c)$ may lead to a state of activation where all one can say about P_n is that there is an instance of P_n involving the entities a, b, and c, but it is not clear which entity fills which role of P_n.

It follows, then, that the depth to which an agent may reason during reflexive reasoning is bounded. Thus an agent may be unable to make a prediction (or answer a query) – even when the prediction (or answer) logically follows from the knowledge encoded in the LTKB – if the length of the derivation leading to the prediction (or the answer) exceeds this bound. It should be possible to relate the bound on the depth of reflexive reasoning to specific physiological parameters, but at this time we are not aware of the relevant data upon which to base such a prediction. We would welcome pointers to appropriate data.

8.3. Nature of inputs to the reflexive reasoner

Our system demonstrates that rulelike knowledge may be used effectively during reflexive reasoning, provided it is integrated into the LTKB and wired into the inferential dependency graph. It also demonstrates that reflexive reasoning can effectively deal with small dynamic input in the form of facts.[33] We suspect that the ability of any reflexive-reasoning system to deal with novel rulelike information will be extremely limited; if the input contains rulelike information that is not already present in the LTKB, the agent may have to revert to a reflective mode of reasoning in order to use this information. This may partially explain why human agents find it difficult to perform syllogistic reasoning without deliberate and conscious effort even though, in a formal sense, such reasoning is simpler than some of the reasoning tasks we can perform in a reflexive manner. In syllogistic reasoning, the "input" has the form of rules and the reflexive reasoner may be unable to use them unless they are already part of the LTKB.

8.4. The reflexive-reasoning system and production systems

As may be evident, there exists a correspondence between a production system and the reflexive-reasoning system described in this article – the declarative memory corresponds to long-term facts, productions correspond to rules, and the working memory corresponds to the WMRR. Thus our system can be viewed as a parallel-production system.

Estimates of the working-memory capacity of production system models range from very small (about seven elements) to essentially unlimited. Our work points out that the working memory of a reflexive processor can contain a very large number of *elements* (dynamic facts in the case of the reasoning system) as long as (1) the elements do not refer to more than (about) 10 entities, and (2) the elements do not involve the same relation (predicate) more than (about) three times. The proposed system also demonstrates that a large number of rules, even those containing variables, may fire in parallel as long as any predicate is not instantiated more than (about) three times (cf. Newell's suggestion [1980] that while productions without variables can be executed in parallel, productions with variables may have to be executed in a serial fashion).

A number of cognitive models are based on the production system formalism; two of the most comprehensive are ACT* (Anderson 1983) and SOAR (Newell 1990). Neurally plausible realizations of these models, however, have not been proposed. Although several aspects of ACT* such as its use of levels of activation and weighted links have neural underpinnings, it has not been shown how certain critical aspects of the model may be realized in a neurally plausible manner. For example, ACT* represents productions with variables, but Anderson does not provide a neurally plausible explanation of how bindings are propagated and how nodes determine whether two bindings are the same. In his exposition of SOAR, Newell has analyzed the time course of neural processes to estimate how long various SOAR operations should take, but he did not suggest how a system such as SOAR may be realized in a neurally plausible manner (see Newell 1990, p. 440). Although a complete mapping of comprehensive systems such as SOAR and ACT* to a neurally plausible architecture still remains an open problem, our system could provide a basis for doing so. In this context, the biologically motivated constraints on the capacity of the WMRR indicated by our system seem particularly significant.

8.5. Reflexive reasoning and text understanding

Several problems will have to be addressed in order to integrate the proposed reasoning system with a comprehensive cognitive system. Some of these problems are discussed in section 10; they include (1) interactions between the reflexive-reasoning system and medium-term memory; (2) how medium-term memory is mapped into long-term memory; (3) how the set of entities in the WMRR changes in a fluid manner; and (4) how distinct modules performing different reflexive processes (e.g., a parser and a reasoner) communicate with one another.

The problem of text understanding is particularly relevant because there exists a rich body of empirical data on the role of inferences based on long-term knowledge during language understanding. The data strongly suggest that certain types of inferences (i.e., inferences that help establish referential and causal coherence) do occur very rapidly and automatically during text understanding (see, e.g., Carpenter & Just 1977; Keenan et al. 1984; Kintsch 1974; McKoon & Ratcliff 1980). The evidence for the automatic occurrence of *elaborative* inferences, however, is mixed (see, e.g., Kintsch 1988; McKoon & Ratcliff 1986; Potts et al. 1988; Singer & Ferreira 1983). Elaborative inferences predict highly likely consequences of events mentioned in the discourse and correspond to forward reasoning in our system. However, as Potts et al. (1988) point out, available experimental evidence does not rule out the possibility that elaborative inferences are performed during reading. The experiments involve two-sentence texts, and it is likely that the subjects do not have any inherent interest in making predictive inferences. It may turn out that subjects do make such inferences when reading longer texts.

Our system suggests that reflexive reasoning can occur in backward as well as forward direction (although, as pointed out in sect. 8.2, there are critical differences in the form of rules that participate in the two types of reasoning). This suggests that agents may perform inferences required for establishing referential and causal coherence as well as predictive inferences in a reflexive manner. The system's prediction can be resolved with the observed data if we assume that the results of predictive inferences only last for a short time (say a few hundred msec) and then disperse unless subsequent input (text) indicates that these inferences are significant and/or relevant to the discourse. Only those inferred facts that turn out to be relevant are encoded in medium-term memory and become available for a longer time.

The extensive body of empirical data on the role of long-term knowledge and inferences in reading will inform future work on our model of reflexive reasoning. At the same time, we hope that the constraints on the form of rules and the capacity of the working memory underlying reflexive reasoning that have emerged from our work will help experimental psychologists in formulating and testing novel hypotheses about the role of reflexive reasoning in reading.

8.6. Reflexive reasoning and the fan effect

Our initial hypothesis as well as our system's behavior suggests that the time taken by reflexive reasoning is independent of the size of the LTKB. This conflicts with the fan effect (Anderson 1983; Reder & Ross 1983); this effect generally refers to the following phenomenon: The more facts associated with a particular concept, the slower the recognition of any one of the facts. We hypothesize that the fan effect applies only to medium-term knowledge and not to long-term knowledge (we use *long-term* in the sense discussed in sect. 1.1). Consider the nature of the task that leads to the fan effect. An agent studies a set of facts until he can recall them. Subsequently, the agent is asked to recognize and make consistency judgments about the learned material and his reaction times are recorded. It is observed that the time taken to recognize a fact increases with the number of facts studied by the agent involving the same concept(s). Observe, however, that the fan effect concerns an arbitrary collection of facts that the agent studied prior to the experiment. We hypothesize that these facts are only encoded in the agent's medium-term memory and are not assimilated into the agent's LTKB. Thus the fan effect is not about facts in the LTKB, rather it is about facts in medium-term memory.

9. Related work

In spite of the apparent significance of reflexive reasoning there have been few attempts at modeling such rapid inference with reference to a large body of knowledge. Some past exceptions are Fahlman's (1979) work on NETL and Shastri's (1988a) work on a connectionist semantic memory (see also Geller & Du 1991). Both these models primarily deal with inheritance and classification within an *IS-A* hierarchy. Hölldobler (1990) and Ullman and van Gelder (1988) have proposed parallel systems for performing more powerful logical inferences, but these systems have unrealistic space requirements. The number of nodes in Hölldobler's system is quadratic in the size of the LTKB, and the number of processors required by Ullman and van Gelder is even higher.[34] A significant amount of work has been done by researchers in knowledge representation and reasoning to identify classes of inference that can be performed efficiently (e.g., see Bylander et al. 1991; Frisch & Allen 1982; Kautz & Selman 1991; Levesque 1988; Levesque & Brachman 1985; McAllester 1990). The results, however, have largely been negative. The few positive results reported do not provide insights into the problem of reflexive reasoning because they assume a weak notion of efficiency (polynomial time), restrict inference in implausible ways (e.g., by excluding chaining of rules), and/or deal with overly limited expressiveness (e.g., only propositional calculus).

9.1. *Relation between* NETL *and the proposed system*

It was pointed out in section 3 that as an abstract computational mechanism temporal synchrony is related to the notion of marker passing. It was also mentioned that Fahlman (1979) had proposed the design of a parallel marker-passing machine (NETL) that could solve a class of inheritance and recognition problems efficiently. But as discussed in section 3, NETL was not neurally plausible. In view of the correspondence between temporal synchrony and marker passing, our system offers a neurally plausible realization of marker passing. It is important to underscore the significance of this realization. First, nothing is stored at a node in order to mark it with a marker. Instead, the time of firing of a node relative to other nodes and the coincidence between the time of firing of a node and that of other nodes has the effect of marking a node with a particular marker. Furthermore, a node does not have to match anything akin to markers. It simply has to detect whether appropriate inputs are coincident. Second, the system does not require a central controller. Once a query is posed to the system by activating appropriate nodes, it computes the solution without an external controller directing the activity of every node at every step of processing. The system's ability to do so stems from the distributed control mechanisms that are an integral part of the representation. Some examples of such built-in mechanisms that automatically control the propagation of activation are the $C_{\uparrow}$, and $C_{\downarrow}$ *relay* nodes in concept clusters (sect. 5.2), and the *switch* networks associated with concepts and predicates that automatically direct the flow of activation to unused banks (sect. 5.2 & 6). Third, our realization of marker passing quantifies the capacity of the working memory underlying reflexive processing in terms of biological parameters. As we have seen, these constraints have psychological significance.

In addition to demonstrating that a marker-passing system can be realized in a neurally plausible manner, our system shows that a richer class of representation and reasoning problems than that realized in NETL can be solved using temporal synchrony – and, hence, marker passing. If we set aside the issue of exceptional knowledge (see below), NETL represented an *IS-A* hierarchy and *n*-ary facts, where terms in a fact could be types or instances in the *IS-A* hierarchy. NETL, however, did not represent rules involving *n*-ary predicates. NETL derived inherited facts by replacing terms in a fact by their subtypes or instances (this characterization accounts for NETL's ability to perform simple [unary] inheritance as well as relational inheritance), but it did not combine inheritance with rule-based reasoning. Consider the example of relational inheritance where *preys-on(Sylvester, Tweety)* is derived from *preys-on(Cat, Bird)*. Observe that this only involves substituting Sylvester for Cat and Tweety for Bird on the basis of the *IS-A* relations *is-a(Sylvester, Cat) and is-a(Tweety, Bird)*. This form of reasoning is weaker than that performed by our system. Our reasoning system can also encode rules such as $\forall x,y$ *preys-on(x,y)* $\Rightarrow$ *scared-of(y,x)*, and given *preys-on(Cat, Bird)* it cannot only infer *preys-on(Sylvester, Tweety)* but also *scared-of(Tweety, Sylvester)*.

The presence of a central controller allowed NETL to compute and enumerate results of queries involving an arbitrary sequence of set intersection and set union operations. NETL's central controller could decompose a query into the required sequence of intersection and union operations and instruct NETL to perform these operations in the proper sequence. This is something our reflexive-reasoning system does not (and is not intended to) do.

NETL also allowed exceptions in the *IS-A* hierarchy, but its treatment of exceptions suffered from serious semantic problems (see Fahlman et al. 1981; Touretzky 1986). In sections 5.4 and 5.5 we described how rules with type restrictions are encoded in our system and explained how this encoding may be extended to deal with type preferences so that the appropriateness – or strength – of a rule firing in a specific situation may depend on the types of the entities involved in that situation. The ability to encode evidential rules will allow our system to incorporate exceptional and default information in an *IS-A* hierarchy (see below).

9.2. CSN: *a connectionist semantic memory*

Shastri (1988a; 1988b) developed CSN, a connectionist semantic network that could solve a class of inheritance and classification problems in time proportional to the depth of the conceptual hierarchy. CSN computed its solutions in accordance with an evidential formalization and dealt with exceptional and conflicting information in a principled manner. It found the most likely answers to inheritance and recognition queries by combining the information encoded in the semantic network. CSN operated without a central network controller that regulated the activity of its nodes at each step of processing. This was the result of using distributed mechanisms (e.g., relay nodes) for controlling the flow of activity. A complete integration of a CSN-like system and the proposed reasoning system should lead to a system capable of dealing with evidential and conflicting rules and facts in a principled manner.

9.3. *Some connectionist approaches to the dynamic-binding problem*

Feldman (1982) addressed the problem of dynamically associating any element of a group of N entities with any element of another group of N entities using an interconnection network. He showed how it was possible to achieve the association task with an interconnection network having only $4N^{3/2}$ nodes. The work, however, did not address how such a representation could be incorporated within a reasoning system where bindings need to be propagated.

Touretzky and Hinton (1988) developed DCPS, a distributed connectionist production system, to address the problem of rule-based reasoning within a connectionist framework. The ability of DCPS to maintain and propagate dynamic bindings is, however, quite limited. First, DCPS can only deal with rules that have a single variable. Second, DCPS is serial at the knowledge level, because each step in its reasoning process involves selecting and applying a single rule. Thus in terms of efficiency, DCPS is similar to a traditional (serial) production system and must deal with the combinatorics of search. Third, it assumes that there is only one candidate rule that can fire at each step of processing. Hence it is not a viable model of reflexive reasoning.

Smolensky (1990) describes a representation of dynamic bindings using tensor products. Arguments and fillers are viewed as *n* and *m* dimensional vectors, respectively, and a binding is viewed as the $n * m$ dimensional vector obtained by taking the tensor product of the appropriate argument and filler vectors. The system encodes arguments and fillers as patterns over pools of *n* argument and *m* filler nodes and argument bindings over a network of $n * m$ nodes. The system can only encode $n * m$ bindings without cross-talk, although a greater number of bindings can be stored if some cross-talk is acceptable. Dolan and Smolensky (1989) describe TPPS, a production system based on the tensor product encoding of dynamic bindings. However, like DCPS, TPPS is also serial at the knowledge level and allows only one rule to fire at a time.

The primary cause of knowledge-level serialism in DCPS and TPPS is that these systems represent arguments and fillers as patterns of activity over common pools of nodes. This severely limits the number of arguments, fillers, and dynamic bindings that may be represented at the same time. In contrast, the compact encoding of predicates, arguments, and concepts in our system allows it to represent and propagate a large number of dynamic bindings simultaneously.

Another system that uses compact encoding and supports knowledge-level parallelism is ROBIN (Lange & Dyer 1989). This system was designed to address the problem of natural-language understanding – in particular, the problem of ambiguity resolution using evidential knowledge. ROBIN and our system have several features in common; for example, ROBIN can also maintain a large number of dynamic bindings and encode "rules" having multiple variables. There are also important differences: ROBIN permanently allocates a unique numerical *signature* to each constant in the domain and represents dynamic bindings by propagating the signature of the appropriate constant to the argument(s) to which it is bound. The use of signatures allows ROBIN to deal with a large number of entities during an episode of reasoning. There is, however, a potential problem with the use of signatures: If each entity has a unique signature, then signatures can end up being high-precision quantities. For example, assigning a distinct signature to 50,000 concepts will require a precision of 16 bits. Hence propagating bindings would require nodes to propagate and compare high-precision analog values. This problem may be circumvented by representing signatures as *n*-bit vectors and encoding arguments as clusters of *n* nodes communicating via bundles of links (see sect. 9.4).

The temporal-synchrony approach can be compared to the signature-based approach as follows: Although the total number of entities is very large, the number of entities involved in a particular reasoning episode is small. Hence instead of assigning a distinct signature to every entity, it suffices to assign distinct signatures to only entities that are participating in an episode of reasoning. Furthermore, this assignment need exist only for the duration of a reasoning episode. One can interpret the relative phase in which a node is firing as such a transient signature of the node. The discussion in section 8.2 about working memory and medium-term memory (also sect. 10) suggests how an augmented system that includes medium-term memory may engage in tasks involving more than 10 or so entities.

Barnden and Srinivas (1991) have proposed Conposit, a connectionist production system. In Conposit, patterns are associated by virtue of the relative position of registers containing these patterns, as well as the similarity between patterns. Argument bindings are propagated by a connectionist interpreter that reads the contents of registers and updates them. We believe that Conposit may be an appropriate architecture for modeling complex reflective processes, but it may not be best suited for modeling reflexive reasoning.

Another solution to the binding problem is based on frequency modulation, whereby dynamic bindings may be encoded by having the appropriate nodes fire with the same frequency (Tomabechi & Kitano 1989).

9.4. Using patterns for propagating bindings

An important aspect of the proposed reasoning system is the organization of *n*-ary rules into a directed graph, wherein the inferential dependencies between antecedent and consequent predicates together with the correspondence between the predicate arguments are represented explicitly. As we have seen, this encoding in conjunction with the temporal representation of dynamic bindings leads to an efficient reasoning system. But the above encoding of rules is significant in its own right. One may take this framework for organizing rules and obtain other organizationally isomorphic connectionist systems by using alternative techniques (e.g., frequency encoding) for representing dynamic bindings. These systems, however, will differ in the size of the resulting network, constraints on the nature of reasoning, reasoning speed, and biological plausibility. To illustrate how the suggested organization of rules and arguments may be combined with alternate techniques for propagating dynamic bindings, we use the proposed encoding of rules in conjunction with what may be referred to as the pattern-containment approach.[35]

In the pattern-containment approach we assume that each argument is represented by a cluster of *n* nodes, and inferential links between arguments are represented by connecting the nodes in the associated argument clusters. An *n*-dimensional pattern of activity is associated with each concept (i.e., an instance or a type), and a dynamic binding between a concept and an argument is represented by inducing the pattern of activation associated with the concept in the appropriate argument cluster. The propagation of dynamic bindings in the system occurs by the propagation (replication) of patterns of activity along connected argument clusters.

It is instructive to compare the pattern-containment approach with the temporal-synchrony approach. The key question is: What is the significance of the pattern of activity that is associated with a concept and propagated across argument clusters? One possibility is that each *n*-dimensional pattern encodes the signature associated with some concept (Lange & Dyer 1989). As we pointed out earlier, the value of n would depend on N, the number of distinct concepts represented in the system. If we assume that concepts are assigned arbitrary patterns as signatures, n would equal log_2N. Alternatively the pattern of activity could encode all the microfeatures of a concept (Hinton 1981; Rumelhart & McClelland 1986). Such a pattern, however, would have to be even larger. Both of these interpretations of patterns make suboptimal use of computational resources: Each argument cluster has to be large enough to encode the full signature of a concept or all the microfeatures associated with a concept. Also, individual bindings have to be propagated by propagating large patterns of activity. An attractive alternative would be to assume that the patterns associated with concepts during the propagation of bindings are some sort of "reduced descriptions." We suggest that the temporal-synchrony approach does exactly this – albeit in an unusual manner. During the propagation of bindings, the relative phase of firing of an active concept acts as a highly reduced description of that concept.

The use of temporal synchrony enables our system to do with one node and one link what the pattern-containment approach does using n nodes and links. The temporal approach also leads to a simple encoding of long-term facts. In contrast, the realization of a long-term fact in the pattern-containment approach will be more complex since it must support mn-bit comparisons (where m is the arity of the fact predicate) to check whether the dynamic bindings match the static bindings encoded in the fact. In section 7.3 we suggested that single (idealized) nodes in our system would have to be mapped to ensembles of nodes and single (idealized) links would have to be mapped to a group of links. This mapping, however, was required to deal with noise in the system and the pattern-containment approach will also have to be augmented in order to deal with noise.

10. Discussion

We have presented a neurally plausible model for knowledge representation and reflexive reasoning. The model supports the long-term encoding of general instantiation-independent structures as well as specific situations involving n-ary relations. It also supports the representation of dynamic information and its interaction with long-term knowledge. Everything presented in this target article, except for the treatment of soft rules (sect. 5.5), has been simulated. The proposed model makes several specific predictions about the nature of reflexive reasoning and the capacity of the working memory underlying reflexive reasoning. These predictions are verifiable and we hope that they will be explored by experimental psychologists. The proposed representational mechanisms are quite general and should be applicable to other problems in cognition whose formulation requires the expressive power of n-ary predicates and whose solution requires rapid and systematic interactions between long-term and dynamic structures. These include problems in high-level vision, other problems in language processing such as syntactic processing, and reactive planning. Below we discuss some problems that need to be addressed if the representational mechanisms proposed here are to be applied in an extended setting.

10.1. Where do phases originate?

In a sense, the "source" of rhythmic activity in the proposed reasoning system is clearly identifiable: The process that poses a query to the system provides staggered oscillatory inputs to entities mentioned in the query and thereby activates them in distinct phases. In a composite perceptual/linguistic/reasoning system, however, such a separation in the phase of the firing of distinct entities must occur intrinsically. For example, the utterance "John gave Mary Book1" should automatically result in the representations of "John," "Mary," and "Book1" firing in different phases and synchronously with *giver*, *recipient*, and *give-obj*, respectively.

The problems of automatic phase separation and consequent segmentation and feature binding has been addressed by several researchers. For example, Horn et al. (1991) demonstrate how an input pattern containing a red square and a blue circle can result in the firing of nodes representing the features "red" and "square" in one phase, and the nodes representing the features "blue" and "circle" in a different phase. The model, however, does not work if there are more than two objects. An internal attentional mechanism similar to the "searchlight" proposed by Crick (1984) may be required for dealing with more elaborate situations.

In the case of linguistic input, we believe that the initial phase separation in the firing of each constituent is the outcome of the parsing process. The parser module expresses the result of the parsing process – primarily the bindings between syntactic arguments and constituents – by forcing appropriate nodes to fire in and out of synchrony. This is illustrated in a parser for English, designed by Henderson (1991), using the proposed model for reflexive reasoning.

10.2. Who reads the synchronous firing of nodes?

There is no homunculus in our system that "reads" the synchronous activity to detect dynamic bindings. Instead, the synchronous activity is "read" by various long-term structures in the system that do so by simply detecting coincidence (or the lack of it) among their inputs. For example, long-term facts read the rhythmic activity as it propagates past them and become active whenever the dynamic bindings encoded in the activity are appropriate. Similarly, τ-or nodes enforce type restrictions (e.g., the node a in Fig. 24) by enabling the firing of a rule whenever the appropriate argument and type nodes are firing in-phase. We have also designed a connectionist mechanism that automatically extracts answers to *wh*-queries and relays them to an output device (McKendall 1991). We associate a code or a "name" with each concept. This name has no internal significance and is meant solely for communicating with the system's environment. The mechanism channels the names of concepts that constitute an answer to an output buffer in an interleaved fashion. For example, the patterns for Ball1 and Book1 would alternate in the output buffer after the *wh*-query *own(Mary, x)?* is posed with reference to the network in Figure 12.

10.3. How are phases recycled?

The constraint that computations must involve only a small number of entities at any given time seems reasonable if we restrict ourselves to a single episode of reasoning, understanding a few sentences, or observing a simple scene. But what happens when the agent is participating in a dialogue or scanning a complex scene where the total number of significant entities exceeds the number of distinct phases that can coexist. In such situations the set of entities in "focus" must keep changing constantly, with entities shifting in and out of focus in a dynamic manner. Identifying the mechanisms that underlie such internal shifts of attention and cause the system's oscillatory activity to evolve smoothly so that new entities start firing in a phase while entities presently firing in a phase gradually "release" their phase remains a challenging open problem (but see Crick & Koch 1990a). In this context one must also note that the notion of an entity is itself very fluid. In certain situations, John may be an appropriate entity. In other situations, John's face or perhaps even John's nose may be the appropriate entity.

The notion of the release of phases has a natural interpretation in the parsing system described by Henderson (1992). The parser is incremental and its output is a sequence of derivation steps that leads to the parse. The entities in the parser are nonterminals of the grammar, and hence each active nonterminal must fire in a distinct phase. Under appropriate conditions during the parsing process – for example, when a nonterminal ceases to be on the right frontier of the phrase structure – the phase associated with a nonterminal can be "released" and, hence, become available for nonterminals introduced by subsequent words in the input. This allows the parser to recover the structure of arbitrary long sentences as long as the dynamic state required to parse the sentence does not exceed the bounds on the number of phases and the number of instantiations per predicate.

10.4. Generalizing the use of synchronous oscillations

Thus far we have assumed that the scope of phase distribution is the entire system. We must, however, consider the possibility where the system is composed of several modules (say the perceptual, linguistic, or reasoning modules). If we combine the requirements of all these modules it becomes obvious that ten or so phases will be inadequate for representing all the entities that must remain active at any given time. Thus a temporal coding of dynamic bindings is not viable if a single phase distribution must extend across all the modules. Therefore it becomes crucial that each module has its own phase distribution so that each module may maintain bindings involving ten or so entities. This, however, poses a problem: How should modules communicate with each other in a consistent manner? Consider a system whose visual module is seeing "John" and whose conceptual module is thinking something about John. How should the visual and conceptual modules share information about John even though the phase and frequency of the nodes encoding John in the two systems may be different? Aaronson (1991) describes a connectionist interface that allows two phase-based modules, each with its own phase structure, to exchange binding information.

10.5. Memorizing facts: Converting dynamic bindings to static patterns

In the proposed system, dynamic information is represented as transient rhythmic activity and long-term memory is encoded by using "hard-wired" interconnections between nodes. We have not discussed how appropriate dynamic information may be converted into, and recorded as, synaptically encoded long-term structures. A specific problem concerns the conversion of dynamic bindings corresponding to a novel (but salient) fact into a medium-term fact by converting the set of dynamic bindings into a set of static bindings that last longer than a few hundred milliseconds (perhaps even days or weeks). This problem has been addressed in Geib (1990) by using recruitment learning (Feldman 1982; Shastri 1988a; Wickelgren 1979) in conjunction with a fast weight-change process abstractly modeled after long-term potentiation (Lynch 1986). The proposed solution allows a one-shot conversion of dynamic facts into a structurally encoded fact in the presence of a "learn" signal. It is envisaged that subsequently, such medium-term structures can be converted into long-term structures by other processes (Marr 1971; Squire 1987; Squire & Zola-Morgan 1991). The notion of fast synapses proposed by von der Malsburg (1981) may also play an intermediate role in sustaining memories that must last beyond a few hundred milliseconds.

10.6. Learning rules

The problem of learning the representation of rules in a system that uses a temporal representation is no more difficult than the problem of learning structured representation in connectionist networks. Instead of being triggered by "simple" coactivation, learning must now be triggered by synchronous activation. Recently, Mozer et al. (1991) have demonstrated how backpropagation style learning may be generalized to networks of nodes that are essentially like ρ-btu nodes. We are addressing the problem of learning in the concept of preexisting predicates and concepts where it is desired that the cooccurrence of events should lead to the formation of appropriate connections between predicate arguments. A special case involves assuming generic interconnections between predicate arguments, and viewing rule learning as learning the correct type restrictions/preferences on argument fillers. This may be achieved by modifying weights on links between the type hierarchy and the rule component (see sects. 5.4 & 5.5).

ACKNOWLEDGMENTS
Thanks to Moshe Abeles, John Barnden, Elie Bienenstock, Gary Cottrell, Mike Dyer, Jerry Feldman, George Gerstein, Pat Hayes, Geoff Hinton, Christopher von der Malsburg, Jordan Pollack, Terry Sejnowski, Paul Smolensky, Simon Thorpe, Dave Touretzky, and several anonymous referees for their comments and suggestions. Thanks to D. R. Mani for his work on the *IS-A*

hierarchy interface and the multiple instantiation problem, and for simulating the model and generating figures. This research was supported by NSF grant IRI 88-05465, ARO grants DAA29-84-9-0027 and DAAL03-89-C-0031, and the DRG grant Schr 275/7-1.

NOTES

1. For example, see Allen 1987; Bobrow & Collins 1975; Charniak 1976; Corriveau 1991; Dyer 1983; Fahlman 1979; Just & Carpenter 1977; Kintsch 1974; Lehnert & Ringle 1982; Norvig 1989; Schank & Abelson 1977; Wilensky 1983.

2. That reflexive reasoning occurs spontaneously and without conscious effort does not imply that the agent cannot become aware and conscious of the result of such reasoning, as in an agent's yes response to the question Does John own a car? given "John bought a Rolls-Royce." In many situations, however, the result of reflexive reasoning may only be manifest in the mental state of the agent. For example, during reading the effect of such reasoning may be manifest primarily in the agent's sense of understanding, coherence (or lack thereof), disbelief, humor, and so forth.

3. The reflexive/reflective distinction we make here in the context of reasoning shares a number of features with the automatic/controlled distinction proposed by Schneider and Shiffrin (Schneider & Shiffrin 1977; Shiffrin & Schneider 1977; see also Posner & Snyder 1975). Like automatic processing, reflexive reasoning is parallel, fast, occurs spontaneously, and the agent is unaware of the reasoning process per se. However, the working memory underlying reflexive reasoning has specific capacity limitations (see sect. 8.2). In formulating the problem of reflexive reasoning and developing a detailed computational model for it, we have generalized the notion of automatic processing by bringing into its fold the more conceptual task of systematic reasoning.

4. If we assume that information is encoded in the firing rate of a neuron then the amount of information that can be conveyed in a "message" would depend on ΔF, the range over which the firing frequency of a presynaptic neuron can vary, and ΔT, the window of time over which a postsynaptic neuron can "sample" the incident spike train. ΔT is essentially how long a neuron can "remember" a spike and depends on the time course of the postsynaptic potential and the ensuing changes in the membrane potential of the postsynaptic neuron. A plausible value of ΔF may be about 200. This means that in order to decode a message containing 2 bits of information, ΔT has to be about 15 msec, and to decode a 3-bit message, it must be about 35 msec. One could argue that neurons may be capable of communicating more complex messages by using variations in interspike delays to encode information (see, e.g., Strehler & Lestienne 1986). However, Thorpe and Imbert (1989) have argued that in the context of rapid processing, the firing rate of neurons relative to their available time to respond to their inputs implies that a presynaptic neuron can only communicate one or two spikes to a postsynaptic neuron before the latter must produce an output. Thus the information communicated in a message remains limited even if interspike delays are used as temporal codes. This does not imply that networks of neurons cannot represent and process complex structures. Clearly they can. The interesting question is how.

5. This observation does not presuppose any particular encoding scheme and applies to localist and distributed, as well as hybrid, schemes of representation. The point is purely numerical – any encoding scheme that requires n^2 nodes to represent an LTKB of size n will require 10^{16} nodes to represent an LTKB of size 10^8.

6. This hypothesis does not conflict with the fan effect (Anderson 1983; see also sect. 8.6).

7. The rules used in this and other examples are only meant to illustrate the dynamic-binding problem and are not intended to be a detailed characterization of commonsense knowledge. For example, the rule relating "giving" and "owning" is an oversimplification and does not capture the richness and complexity of the actual notions of giving and owning.

8. Although *systematicity* has broader connotations (e.g., see Fodor & Pylyshyn 1988a), we use it here to refer specifically to the correspondence between predicate arguments stipulated by rules.

9. The symbol $\forall$ is the universal quantifier, which may formally be interpreted to mean "for all," and the symbol $\Rightarrow$ is the logical connective "implies." Thus the statement $\forall u,v\ [buy(u,v) \Rightarrow own(u,v)]$ asserts that for any assignment of values to u and v, if u buys v then u owns v.

10. A similar formation of "static" bindings occurs in any learning network with hidden nodes. Observe that a hidden node at level l learns to respond systematically to the activity of nodes at levels $l-1$ and below, and in so doing the network *learns* new bindings between representations at level l and $l-1$. These bindings, however, are static, and the time it takes for them to get established is many orders of magnitude greater than the time within which dynamic bindings must be established.

11. Feature binding can be achieved by creating sets of features such that those belonging to the same entity are placed in the same set. In terms of expressive power, *unary* predicates suffice to solve this problem. For example, the grouping of features belonging to a "red smooth square" and a "blue dotted circle" can be expressed by using unary predicates such as $red(obj1) \wedge smooth(obj1) \wedge square(obj1)$ and $blue(obj2) \wedge dotted(obj2) \wedge circle(obj2)$.

12. We first described our proposed model in 1990 (Shastri & Ajjanagadde 1990). An earlier version using a central clock was reported in Ajjanagadde and Shastri (1989).

13. As stated in Note 11, *unary* predicates suffice to solve the feature-binding problem and the expressive power of the models cited above is limited to *unary*-predicates (see Hummel & Biederman 1991). The greater expressive power provided by n-ary predicates would eventually be required by more sophisticated models of visual processing.

14. There are other variants of marker passing (see, e.g., Charniak 1983; Hendler 1987; Hirst 1987; Norvig 1989) where "markers" are even more complex messages containing a marker bit, a strength measure, backpointers to the original and immediate source of the marker, and sometimes a flag that indicates which types of links the marker will propagate along. The marker-passing system has to process the information contained in markers, extract paths traced by markers, and evaluate the relevance of these paths. In view of this, such marker-passing systems are not relevant to our discussion.

15. We can generalize the behavior of a ρ-btu node to account for weighted links by assuming that a node will fire if and only if the weighted sum of synchronous inputs is greater than or equal to n (see sects. 5.5 & 8.1).

16. In the idealized model each argument is encoded as a single ρ-btu node and, hence, it is reasonable to assume that a node may fire in response to a single input. The thresholds of nodes in the ensemble-based model will be higher and will depend on the average interensemble connections per node.

17. A constant refers to a specific entity in the domain, the symbol $\exists$ is the existential quantifier, which may be interpreted to mean "there exists." Recall that the symbol $\forall$ is the universal quantifier, which may be interpreted to mean "for all." Thus the statement $\forall x\ [person(x) \Rightarrow \exists z\ mother(z,x)]$ asserts that for every person x there exists some z such that z is the mother of x. The symbol $\wedge$ is the logical connective "and."

18. The system can encode first-order, function-free Horn Clauses with the added restriction that any variable occurring in multiple argument positions in the antecedent of a rule must also appear in the consequent. Horn Clauses form the basis of PROLOG, a programming language used extensively in artificial intelligence (see, e.g., Genesereth & Nilsson 1987).

19. This time consists of (1) $l\pi$, the time taken by the activation originating at the enabler of the query predicate to reach the enabler of the predicate(s) that are relevant to the derivation of the query, (2) π, the time taken by the relevant fact(s) to become active, (3) π, the time taken by the active fact(s) to activate the relevant collector(s), and (4) $l\pi$, the time taken by the activation to travel from the collectors of the relevant predicate(s) to the collector of the query predicate.

20. The closed-world assumption simply means that any fact *F* that is neither in the knowledge base nor deducible from the knowledge base may be assumed to be false.

21. Here we are using *concept* to refer only to the entities and types encoded in the hierarchy. This is not to suggest that predicates such as *give* and *own* that are not represented in the *IS-A* hierarchy are not concepts in the broader sense of the word.

22. In our formulation, each *IS-A* link is strict and only property values are exceptional. This approach for dealing with exceptional and defeasible information in *IS-A* hierarchies is explained in Shastri (1988a).

23. This is required because a fact is true of some entity of type *C* if one or more of the following holds: (1) The fact is universally true of a superconcept of *C*, (2) the fact is true of some subconcept/instance of *C*, or (3) the fact is universally true of a superconcept of a subconcept/instance of *C*. The last is required if concepts in the *IS-A* hierarchy can have multiple parents.

24. These times are approximate because the time required for propagation along the *IS-A* hierarchy and the rules may overlap and, hence, the actual time may be less. For example, the time to perform a predictive inference may also only be $max(l_1\pi, 3l_2\pi)$. It is also possible for the actual time to be greater, because in the worst case it may take up to eight cycles instead of three to traverse an *IS-A* link.

25. The number of antecedent predicates (*m*) in a rule can also be reduced by introducing ancillary predicates. For example, the rule $\forall x,y,z\ P(x, y, z) \wedge Q(x, y, z) \wedge R(x, y, z) \Rightarrow S(x, y, z)$ may be replaced by two rules, each of which has only two antecedent predicates: $\forall x,y,z\ P(x, y, z) \wedge Q(x, y, z) \Rightarrow S1(x, y, z)$ and $\forall x,y,z\ S1(x, y, z) \wedge R(x, y, z) \Rightarrow S(x, y, z)$. The benefit of reducing *m* in this manner has to be weighed against the cost of introducing an additional predicate in the system. But the savings outweigh the costs if such a predicate helps in reducing the *m* value of several rules.

26. The reasoning system uses the phase of activation to encode binding information. Hence, in principle the amplitude of activation could be used to represent the "strength" of dynamic bindings and rule firings. Note however, that the amplitude of a node's output is encoded by the spiking frequency and the use of varying frequency to encode rule strengths will interfere with the encoding of dynamic bindings.

27. While the occurrence of synchronous activity is less controversial, the occurrence of synchronized oscillations in the animal brain and its representational significance is still a matter of controversy. More evidence is needed to establish firmly the role of oscillatory activity in neural information processing. Some researchers have reported difficulty in demonstrating oscillatory activity in the primate visual system using static stimuli (e.g., Rolls 1991; Tovee & Rolls 1992). In this context, however, it must be recognized that a very small fraction of neurons would be expected to participate in an episode of synchronous activity. Furthermore, the grouping of neurons would be dynamic and vary considerably from one episode of reasoning to another. Hence, synchronous oscillations would be very difficult to detect.

28. A more detailed model of such coupling has since been developed (Mandelbaum 1991).

29. These timings were obtained by analyzing the simulations of the reflexive-reasoning system carried out using a simulation system developed by Mani (1991). The simulation system is implemented using RCS – the Rochester Connectionist Simulator (Nigel et al. 1989).

30. The above behavior generalizes the notion of a "strength" associated with concepts (cf. Anderson 1983) and extends it to rules, *IS-A* relations, facts, and even individual static bindings in the LTKB.

31. The cost of realizing multiple instantiation of concepts is considerably lower than that of realizing the multiple instantiation of predicates. Thus the value of k_1 can be higher than three. Observe however, that k_1 need be no more than $\lfloor \pi_{max}/\omega \rfloor$.

32. There are several ways of encoding the relevant kinship knowledge. All these pose the same problem, however: The antecedent of one of the rules contains a repeated variable that does not occur in the consequent. One possible encoding of the relevant knowledge is given below (note that *Self* refers to the agent and the rest of the names have been chosen arbitrarily to complete the example). The long-term facts are *grandfather(George, Self)*, *mother(Susan, Self)*, and *father (George, Susan)*. The rule is $\forall x,y,z$ *grandfather(x,y)* $\wedge$ *father(x,z)* $\wedge$ *mother(z,y)* $\Rightarrow$ *maternal grandfather(x,y)*.

33. In addition to the constraints on the WMRR, the number of dynamic facts that can be communicated to an agent at one time will be bounded by the rather limited capacity of the overt short-term memory.

34. Ullman and van Gelder (1988) treat the number of nodes required to encode the LTKB as a fixed cost; hence they do not refer to its size in computing the space complexity of their system. If the size of the LTKB is taken into account, the number of processors required by their system turns out to be a high-degree polynomial.

35. The relation between our approach and the pattern-containment approach was pointed out by Geoff Hinton (personal communication).

Open Peer Commentary

Commentary submitted by the qualified professional readership of this journal will be considered for publication in a later issue as Continuing Commentary on this article. Integrative overviews and syntheses are especially encouraged.

Time phases, pointers, rules and embedding

John A. Barnden
Computing Research Laboratory & Computer Science Department, New Mexico State University, Las Cruces, NM 88003-0001
Electronic mail: *jbarnden@nmsu.edu*

Binding by time phases is an interesting special case of the following very general (temporary) binding method: To bind two things, mark them in roughly the same way. Let's call this the similar-mark approach. Note that it could apply to nonconnectionist as well as connectionist systems. In Shastri & Ajjanagadde's (S&A's) case, we may take the marks to be the oscillatory patterns of excitation acquired by argument nodes and so on. Two marks are similar enough to constitute a binding if they have sufficiently similar phases (and frequencies). So, S&A's method is a special case of the approach of temporarily binding connectionist nodes or subnetworks together by dynamically making them hold activation patterns that are similar enough in some specific sense. That is, the time-phase method is a special case of "pattern-similarity association" or PSA (Barnden & Srinivas 1991). This is in turn a connectionist special case of the similar-mark approach.

A benefit of considering the time-phase method in the context of PSA and similar-mark binding in general is that we see the close relationship to the technique of "associative addressing" widely used in specialized computer hardware as an alternative to pointers. With this technique two memory areas can be temporarily "linked" together by placing identical or sufficiently similar bit-strings somewhere within them. Such a bit-string extracted from one place can be used to find the other place or places that contain that bit-string or suitably similar ones. In sum, S&A's system, which is one of the few connectionist systems that can actually perform inferencing of any respectable complexity, turns out to rest on a binding scheme quite strongly related to a conventional computer technique, but without using any close analogue of pointers.

We are being led here to the question of what happens to the notion of a pointer when we move away from computers. This question is examined to some extent by Barnden and Srinivas (1991). One can define a pointer in a connectionist system to be a temporary system substate (e.g., activation pattern) that identifies a permanently existing place in the system. However, without specific architectural assumptions we cannot say what a "place" is, other than by resting on the excessively restrictive option of a place just being a *single* node or on the excessively loose option of a place being *any subset* of the system's nodes. Going back to S&A, if the phase assigned to the John node or assembly, say, were fixed for all time, then phases could be regarded as pointers, because they would permanently identify such nodes or assemblies. However, S&A allow phases to be dynamically assigned, so they are probably radically different from "pointers" under any usefully narrow construal of that word. The signature scheme in ROBIN (Lange & Dyer 1989) is more pointerlike, because signatures are statically assigned.

One major benefit of similar-mark techniques is that they allow bidirectional binding in two senses. (1) A binding could be *conceptually* bidirectional; one might, for instance, say that if several S&A argument nodes have the same phase they are bidirectionally bound to each other. (2) A binding, though perhaps conceptually unidirectional, could be used bidirectionally. For instance, a node oscillating at a certain phase might broadcast its oscillation to other nodes, thereby causing similar-phased nodes to light up in some special way, but the same thing could be done starting at any of those nodes (see also Touretzky 1990). By contrast, computer pointers can only efficiently be used in one direction.

A disadvantage of many similar-mark schemes, however, is that if a binding is conceptually *uni*directional, one needs something extra to specify direction. That something could be highly implicit in the overall system architecture; thus, the binding between an S&A argument node and the John node is, arguably, conceptually unidirectional, and that fact is implicitly respected in the whole way that the system operates. However, if one needed unidirectional bindings between argument nodes for some reason, one would need to do something more than simply give them the same phase.

A concern I have about many connectionist systems, including S&A's, is that they may face difficulty in encompassing certain important types of reasoning, including some reflexive types. S&A claim it is unlikely that the input propositions to reflexive reasoning episodes can be dynamically arising rules. I take the point of their syllogism example, but there are more mundane examples that are not so easily disposed of. For instance, suppose someone says, "All the people at the party were toothbrush salespersons. Some of them even had their sample cases with them." The obvious inference that those cases contained toothbrushes seems no less a candidate for being dubbed "reflexive" than do the inferences in S&A's Little Red Riding Hood and Colombian drug enforcement agency example. Yet one of the input propositions is the universally quantified one about all the people in the room, and this can be viewed as a *dynamically arising rule*.

Now suppose someone says, "Tom thought that the milk was sour. He went out to buy some more." One needs to be able to apply the general knowledge that sour milk tends to be unusable for certain purposes, together with Tom's reported thought, in order to understand Tom's motive in going out for more milk. Is such reasoning not reflexive? That is, I am suggesting that input propositions and reasoning episodes involving them can be embedded in propositional attitude contexts (among other sorts of context, such as counterfactual ones) without making the reasoning nonreflexive. This makes the task of connectionistically implementing reflexive reasoning yet more complex.

Embedding and dynamically arising rules are discussed further in Barnden (1992, pp. 149–78). Because very few workers in connectionism, or indeed critics of connectionism, have even paid lip service to the issues, my comments are hardly a strike against S&A specifically.

Plausible inference and implicit representation

Malcolm I. Bauer

Department of Psychology, Princeton University, Princeton, NJ 08544-1010
Electronic mail: *malcolm@clarity.princeton.edu*

Shastri & Ajjanagadde's (S&A's) distinction between reflexive and reflective reasoning is similar to the distinction between implicit and explicit reasoning made by Johnson-Laird (1983). Implicit reasoning is rapid, effortless, and occurs outside conscious awareness. It is basically model building without the deliberate search for alternatives. Explicit reasoning requires deliberate, conscious effort and calls for the search for alternative models that may invalidate an inference. S&A's work is hence an attempt to create a detailed account of implicit reasoning. They accordingly write that their system "simulates the behavior of the external world and dynamically creates a vivid model of the state of affairs resulting from the given situation" (sect. 3.4). This is an important and innovative area of research but there are some weaknesses in their theory.

First, S&A avoid the important question of how people make a limited number of plausible inferences from the vast set of possible inferences. From any premise, there follow an infinite number of valid deductions and inductive hypotheses (for example, continually conjoining the premise with itself generates a countably infinite set of valid deductions). The mechanisms by which people reason must greatly constrain the inferences they draw. For example, Johnson-Laird (1988) proposes that when people make deductive inferences they draw conclusions that maintain the semantic information in the premises, and when they make inductive generalizations they draw conclusions that increase semantic information. S&A avoid the issue by hand coding only those inference patterns they judge to be plausible. Given the premise "John gave Mary book1" S&A decide that "Mary owns book1" and "Mary can sell book1" are two plausible inferences. However, the constraints on which inferences are drawn are not part of their theory, but are in the heads of the researchers.

Second, S&A's notion of "model" is problematic. Models capture the structure of the world. One can perform analogous operations on models and make inferences about the state of the world from the new state of the model. An important property of models is their ability to represent information implicitly. In so doing, models can represent whole classes of inferences that can be made explicit, if needed, with further reasoning. This kind of representation is quite different from encoding selected plausible inference patterns as in S&A's system. S&A imply that a model is a finite collection of plausible inferences that may be drawn about a situation. When a situation is partially described, the system constructs a chain of inferences. For example, from the premise "John drove from his house to the store" some plausible inferences that follow are, "John left his house" and "John arrived at the store." Equally plausible, though, are inferences such as "John drove at least halfway to the store," "John drove at least ¾ of the way to the store," and "John drove at least ⅞th of the way to the store," and so on. In S&A's system, a complete model of the situation would consist of an explicit representation of all of the above inferences and an infinity of other inferences that follow. [See also *BBS* multiple book review of Sperber & Wilson's *Relevance: Communication and Cognition, BBS* 10(4)1987.] Although it is clear that people can determine the validity of these inferences, it is unlikely that they construct explicit representations for each of the potential inferences. A model of the premise "John drove from his house to the store" is not a series of plausible inferences that follow from it, but rather a representation that captures the structure of John driving from his house to the store. Although an infinite number of inferences follows from this model, reflexive reasoning does not involve constructing all the plausible inferences; rather, it entails constructing a model that represents the situation from which relevant inferences could be drawn as necessary.

In summary, Shastri & Ajjanagadde must overcome these two problems (the lack of a theory of plausible reasoning in the inference mechanism and the inability to represent information implicitly) to make their theory a more credible account of human reflexive reasoning.

Could static binding suffice?

Paul R. Cooper

Institute for the Learning Sciences, Northwestern University, Evanston, IL 60201

Electronic mail: *cooper@ils.nwu.edu*

Dynamic variable binding is widely accepted as a serious challenge for connectionists. Shastri & Ajjanagadde (S&A) have more than met that challenge here: This is an elegant proposal with appealing performance characteristics (e.g., independence of the size of the knowledge base) and equally appealing compatibilities with results from psychology and neuroscience. But the endeavor of addressing the challenge directly, as well as the character of the resulting solution, provokes the desire to reconsider alternatives to a frontal attack on variable binding and rule-oriented reasoning.

If S&A's solution works and is even elegant, why bother to worry about whether there is more to consider? First, possibly the most interesting question for the connectionist enterprise is this: How much can be done in parallel? The traditional connectionist approach when cross-talk appears inevitable is to share net resources in time. Although S&A's solution can hardly be construed as "sequential," it does exploit some time sharing. Improvements may be possible.

Second, S&A's contribution is motivated in large part by a desire to provide a connectionist explanation for traditional rule-oriented reasoning. There are two dangers here. What if rule-oriented reasoning turns out to be unimportant? It is at least conceivable that an alternative paradigm such as case-based reasoning (Riesbeck & Schank 1989) may be more useful, with dynamic variable binding possibly irrelevant. Another possibility is that although some rule-oriented reasoning may be necessary, a full-fledged treatment of *n*-ary predicates is unnecessary and counterproductive, therefore, much more restricted mechanisms may be sufficient.

Constraints and feasibility. So-called static binding may suffice to explain most reflexive reasoning, despite appearances to the contrary (sect. 2.1.1). The essence of the binding problem is associating an argument and filler, or variable and value in the general case. The static-binding solution requires the existence of a unit for *every* feasible pairing of variable and value. Such "binder" units and their connections are stable long-term features of the network; thus, "static." However, it is their *activation* that indicates the presence of an actual binding. Thus, the static-binding solution is in general capable of dynamically associating variables and values during the course of a computation.

Knee-jerk objections to this idea, motivated by the counterintuitive nature of unit/value connectionism (Feldman & Ballard 1982) and the seeming inability of a static structure to support universal, general-purpose problem solving, can be discounted. Even the most simplistic scheme, allowing each of the n entities in memory to associate with any of the others, requires only $O(n^2)$ nodes. But is even n^2 too large? It is on this point that even careful connectionists seem to run aground; indeed, this is apparently one reason why Shastri, who in earlier work exploited static binding (1988a), developed the current work. The value of n can certainly be expected to be 10^5 or better, which makes n^2 too large compared to the available resources in the brain.

This analysis is too simplistic, however. Static binding requires only that we use a unit for every feasible pairing of variable and value, or argument and filler. If feasible values for variables are restricted by exploiting any kind of type or category knowledge (see also sects. 2.4 & 5.4) and if binder units are only allocated for feasible values, the number of required units reduces dramatically. Cooper and Swain (1992) work this idea out in some detail, for example, for a massively parallel implementation of arc consistency. If we limit the number of values per variable to a reasonably large but fixed maximum, the total number of binder units required is linear in the number of variables in memory. In other words, it would not be unreasonable to assume that the node requirement of static binding is at least close to linear in the size of the knowledge base, and certainly much less than n^2. Hence static binding can support the basic task of associating simple variables and values.

The feasibility of static binding becomes less obvious given the necessity to reason about combinations or compositions of simple primitives. On one hand, it is obvious that any value-based encoding just cannot represent all the potential combinations, of arbitrary order, that may occur. On the other hand, solutions exploiting only low-order combinations, particularly pairs (e.g., Feldman 1985), may suffice to explain the simpler tasks solved in "reflexive" as opposed to "reflective" or overtly sequential reasoning (which would be an interesting result in itself). Complex tasks apparently requiring "systematicity" and "composition," such as the recognition of structurally composed objects, are also achievable in this way (e.g., Cooper 1992). Finally, the propagation of bindings hardly poses an insurmountable problem. It is exactly the propagation of constraints that forms the basis of the relaxation process used by most connectionist networks.

Possibly the subtlest issue concerns representing truly novel associations. Clearly, associations exist that a restricted static-binding network cannot represent. Representing such associations requires structural changes to the network learning. But hard learning of entirely new concepts is hardly "reflexive" reasoning. It requires time, repetition, attention, reflection, and so on. Thus, it seems reasonable to assume that such hard structural learning may require special-purpose neural machinery.

To summarize, it is tempting to suppose that Shastri & Ajjanagadde have developed the "last word" on variable binding, and that this irksome challenge can at last be put to rest. But the possibility of simpler, faster, and more parallel methods has not yet been ruled out. The overall point here is hardly irrelevant. That is, if we ignore the cries of variable-users to explain how to replicate their results and we try hard to construct systems that solve hard AI problems while avoiding the obvious exploitation of arbitrary high-order combinations and frequent binding of novel values to variables, we may end up with some very interesting results indeed.

From symbols to neurons: Are we there yet?

Garrison W. Cottrell
Computer Science & Engineering Department 0114, University of California, San Diego, La Jolla, CA 92093
Electronic mail: *gary@cs.ucsd.edu*

Shastri & Ajjanagadde's (S&A's) target article proposes two main ideas, the direct embodiment of a subset of formal logic as structured associations between predicates, and the solution to the binding problem through phase labeling of entities. This is a truly novel solution to the binding problem: The phase-labeling approach avoids the trap of combinatorial space requirements implied by systems that "connect" the two entities through a path in a network. When I first saw this work, for a fleeting moment, I wanted to become a localist again.

Having caught myself at the brink, my critique will focus on the logical inference side of the system. The notion of embodied inference is not novel (Cottrell 1985; 1989; Hebb 1949; James 1890; Lange & Dyer 1989; Shastri 1988b; Touretzky & Hinton 1988), but the current system claims to achieve better efficiency and coverage. I will consider three aspects: technical adequacy, neurological plausibility, and finally, the pattern-containment alternative.

Technical adequacy. It is unclear from the target article whether the quite complicated specific wirings used by this model correctly *implement the inferences* S&A say they do. For a formal system, one usually wants to prove soundness (that only inferences that are entailed by the facts can be derived).[1] Correspondence with the first author has allayed several of my fears, but the burden is on the authors to prove that this complex system does not make incorrect inferences. The matter cannot be decided by inspection.

A second worry is the lack of expressiveness. It appears that negation cannot be represented, as it has not even been mentioned in the text. To address this, for every predicate P, one can simply add another node (or, in this case, a set of nodes) to represent the predicate $\sim P$. Inferences involving $\sim P$ are then driven by activation from this node. A consistency gadget between P and $\sim P$ can be constructed to enforce that they do not both fire when the network has settled. This is the solution used in the Spock system (Cottrell 1985; 1989), which implements Reiter's Default Logic for inheritance (Etherington & Reiter 1983). In S&A's system, the consistency gadget will also have to enforce that the bindings of the two representations of the predicate are the same when enforcing consistency (Spock is propositional).

S&A's claim that they can add defaults (sect. 5.5) and combine the forward and backward systems (sect. 3.5) merits further inspection. Care must be taken that added expressiveness not detract from the efficiency of the system. In the Spock system, I found a classic example of the expressiveness/tractability trade-off (Levesque & Brachman 1985). When the ability to compute the contrapositive is included in the representation (adding $P \rightarrow Q$ implies $\sim Q \rightarrow \sim P$ is encoded also) there exist sets of facts where mixtures of defaults and first-order rules cause long settling times for the network even when there is only one consistent extension. This was due to the backward system inferring $\sim P$'s (begun by a short default inference) while the forward system was inferring P's through the same long chain from the other end. Many iterations were necessary to resolve the conflict. However, in a version where activation was allowed to spread more permissively (dubbed Dr. Spock), settling times were more reasonable due to *leakage* of consistency information through the network. The use of graded evidence in the current system to pick the best solution could alleviate this potential problem.

Finally, one wonders whether the benefits of being able to express propositions about types (sect. 5) is worth the cost of adding another system (the *IS-A* hierarchy) when inheritance can be expressed as repeated applications of logical inference (Hayes 1977).

Neuroscientific plausibility. From the point of view of modeling the brain, this architecture requires a suspension of disbelief about how such a system could have been produced by evolution. This criticism is weak because it is founded on an argument from lack of imagination. However, in order for this system to work properly, highly specific connections must be formed from the nodes representing the concepts to the connections between antecedents and consequents. Things get more complicated when one wants to introduce constants as in Figure 14, or multiple arguments as in Figure 15. Learning in such structured networks uses a recruitment rule (Valiant 1988), where preexisting connections between the appropriate units gain force. In S&A's case, there must be preexisting connections from every potential concept to connections from every potential predicate to fact nodes, which leads to another combinatorial explosion.

The pattern-containing inference alternative. The possibility of a system of pattern-containing inference (sect. 9.4) is a useful one to pursue. This is the idea that a system of embodied inferences like the one proposed in the target article could be constructed that passes distributed patterns of activation from antecedent to consequent in slots for the arguments. It is clear that pattern-containing inference could also use the phase-labeling mechanism to maintain multiple separate bindings for the same pattern.

There are interesting advantages to using pattern-containing embodied inference rules: (1) It should be possible to learn rules using back propagation or some similar technique between antecedents and consequents; (2) *semantic filters* would be embodied in the associations: The copy of predicate arguments from antecedent to consequent is essentially through an autoencoder network (e.g., as in Hanson & Kegl 1987), thus only patterns similar to the ones that have appeared in exemplars would be allowed through. Semantic restrictions on arguments would therefore be handled completely locally, based on experience with this inference, rather than on constraints that must be enforced by connections from the *IS-A* system, as S&A propose. This is inherently more efficient. Also, covariance constraints between arguments would naturally be enforced.

Whether or not one rejects Shastri & Ajjanagadde's system for the above reasons, it is an undeniable achievement of this work that it has brought to light a bold new idea for solving the binding problem with processes available in the brain. The notion of phase labeling of entities is a powerful one, and here for the first time we have a demonstration of its viable use.

NOTE

1. Completeness (that all sound inferences *can* be derived) is not at issue here.

Making a middling mousetrap

Michael R. W. Dawson and Istvan Berkeley
Biological Computation Project, Department of Philosophy, University of Alberta, Edmonton, Alberta, Canada T6G 2E9
Electronic mail: *mike@psych.ualberta.ca*

Emerson once noted that "if a man . . . make a better mousetrap than his neighbor, though he build his house in the woods, the world will make a beaten path to his door." Shastri &

Ajjanagadde (S&A) must clearly be expecting a lot of company at their house in the woods, for they not only believe that they have built a better mousetrap – an effective and systematic reasoner – they also believe that this mousetrap is of a different kind – a rule instantiating, biologically plausible connectionist device. A closer scrutiny of their model reveals, however, that not only is it the same type of mousetrap that classical AI has been springing for decades, but that it is also not quite as good.

Three different weaknesses suggest that the proposed model is classical in nature. First, the target article argues that the reflexive reasoner does not require a central controller, in contrast to classical systems. This cannot be true. The system can only be understood as providing a yes/no answer to questions like, "Can Mary sell Book1?" by having an external controller that understands (and remembers) the original question, and also knows that a response will be encoded as activity in *c*:can-sell. The system cannot autonomously make sense of the blooming, buzzing confusion of its own activity. For instance, Figure 13 illustrates that at one point in time the nodes *c*:can-sell, *c*:own, and *c*:give (representing different possible answers), and the nodes *e*:can-sell, *e*:own, and *e*:buy (representing different possible questions) are all simultaneously active. As a result, the network cannot "know" what answer it is giving, nor the original question that was posed, without external interpretation.

Second, the target article argues that the reflexive reasoner is not a classical system because it is rule instantiating. This amounts to a standard connectionist claim that network architectures do not clearly demarcate processes from data structures (Dawson & Schopflocher 1992). This claim is clearly not true of the proposed model: The rules governing system inferences are qualitatively different and are represented separately from the data structures being processed, as S&A conveniently illustrate with the "squiggly line" in Figure 19.

Third, it is claimed that the reflexive reasoner – unlike classical systems – is biologically plausible. This too is far from established. Although it is quite interesting that temporal synchrony has been observed in the cortex, many more specific claims are not defended in the target article. Several different and highly specific neural circuits are proposed (e.g., Figures 14, 21, 23, 25). In addition, a number of qualitatively different processing units – including three different kinds of tau-or units – are required. A great deal of further evidence from neuroscience is needed to support such claims.

Biological plausibility is further weakened in the context of speculations about how the network might learn facts or rules. If the learning of facts requires the presence of an external "learn" signal, then this strongly indicates that the network is not autonomous and thus is far from being neurally plausible. In addition, although it may be true that the learning of rules in the reflexive reasoner is no more difficult than learning in nontemporal connectionist systems, it is certainly biologically implausible – particularly if backpropagation is used (see Grossberg 1987).

The three arguments above suggest that the proposed model is *not* a mousetrap of a different kind. But is it a *better* mousetrap? Although S&A have shown that a number of the functions found in traditional reasoning systems can be implemented by their novel parallel architecture, their network suffers from some severe logical limitations. Regrettably, these serve both to compromise its inferential power and to cast further doubt upon its putative biological plausibility.

The proposed model has two significant difficulties in dealing with variables that occur in multiple argument positions. First, in backward reasoning it cannot use rules in which a variable occurs in multiple argument positions in a rule's antecedent when the variable does not also appear in the rule's consequent. Second, in forward (or predictive) reasoning, rules in which variables occur in multiple argument positions in the consequent cannot be used unless those variables are also present in the antecedent of the rule and are bound in the reasoning process. Although these two limitations are acknowledged in the target article (sect. 8.2.5), S&A fail to note the full extent of the problems they produce (e.g., with respect to reflexivity).

These are not the only logical difficulties from which the system suffers. For example, the proposed *IS-A* hierarchy cannot handle facts or queries in which existential quantifiers fall within the scope of universal quantifiers. More significant, the network has considerable difficulties in handling multiple instantiations of the same predicate (it also has some lesser difficulties with multiple instantiations of the same concept).

To provide a convincing solution to the latter problems, a considerable number of additional nodes would be required. Unfortunately, this would slow the system's performance significantly. As a result, S&A compromise and limit the number of multiple instantiations of predicates to just three. However, they do not offer any evidence that this limit is either biologically or psychologically plausible. As this limit is critical for calculating the latency of network operations, their claims about the biological plausibility of the system's speed must be treated with a degree of suspicion.

This difficulty, considered in conjunction with the other logical limitations of their network, gives grounds for believing that the proposed model is in fact less powerful than traditional systems. For example, none of the logical limitations mentioned above affect the classical backward-reasoning system described by Pelletier (1982). In short, Shastri & Ajjanagadde have not built a better mousetrap – perhaps they should get a cat!

ACKNOWLEDGMENT
This work was supported in part by NSERC operating grant 2038.

Reasoning, learning and neuropsychological plausibility

Joachim Diederich
Neurocomputing Research Centre, School of Computing Science, Queensland University of Technology, Brisbane Q 4001, Australia
Electronic mail: *joachim@fitmail.fit.qut.edu.au*

Shastri & Ajjanagadde's (S&A's) approach is remarkable in many ways. They offer efficient reasoning in a connectionist knowledge representation system. This representation system has an expressiveness that facilitates the realization of a number of knowledge structures (frames, scripts, etc.). Furthermore, they present a model that makes a number of predictions about psychological processes and therefore allows experimental verification (or falsification). But most important, S&A attempt to close the gap between high-level reasoning and neural processing and show how "reflexive" inferences can be drawn efficiently with slow neural elements.

It is natural that a model that covers a wide range of phenomena cannot be equally specific and appropriate in every detail. A few points that deserve attention should be mentioned.

Learning. S&A do not provide an answer to the question of learning. Although this seems to be a research strategy that is generally accepted in the AI community, the lack of learning is a problem for S&A (sections 10.5 & 10.6 speak about learning in very vague and general terms). In their model, reasoning is based on complex, domain-dependent networks and it is an important question how these network structures are generated. In particular, if S&A extend their claim of neural plausibility to learning, they should answer the question of how complex network structures can be generated based on biologically plausible, namely sparsely connected, neural networks (Diederich 1992; Stevens 1989). Furthermore, although in their model limited reasoning can be done efficiently, the generation of the networks that allow these reasoning processes can be complex or even hard.

Recent neurobiological findings have shown that there are considerable changes in the receptive field organization of cortical cells in adult cats and primates (cf. Merzenich et al. 1988). These changes are triggered by the absence of input (Gilbert & Wiesel 1992; Merzenich et al. 1988) or experience (e.g., tactile stimulation, Merzenich et al. 1988). These processes are relatively fast. The modification of the receptive field organization of cells can be noticed within a few minutes. Gilbert and Wiesel (1992, p. 152) assume that dynamic changes in receptive field structure may occur continuously during normal vision.

On the connectionist level, these processes are modeled by "recruitment learning" methods. Adding learning techniques such as recruitment learning to S&A's model can help to replace some artificial components of the system, for example, the switching devices and the allocation of free memory banks for multiple instances, with biologically plausible, structure-changing learning methods.

Grounding. Over the past several years several authors (e.g., Pfeifer & Verschure 1992) have pointed out that we cannot understand a cognitive system without a connection to sensory experience. That is, mental states (instantiated predicates in S&A's system) develop out of real interactions with the physical world. The important point is that it is not sufficient to link a high-level reasoning system with a sensory system in order to realize such a connection, but that the conceptual representation itself is the result of interactions with the environment and includes sensory pathways that are at least partially modified by experience. A connectionist reasoning system without any learning does not allow concept formation in this sense and is therefore restricted as a psychological model.

Neural plausibility. Although some aspects of the model are biologically plausible (synchronous activity, fast response times, etc.), some components are purely functional elements. The "enabler" and "collector" units as well as the "τ and" and "τ-or" units are used to allow reasoning and are not immediately plausible neural elements. The same holds for the connectivity pattern for long-term knowledge base (LTKB) facts. As a matter of fact, the unit types and network structures are excluded from the discussion of biological plausibility (section 7 speaks about "synchronous, rhythmic activity" only). It is possible to find neurobiological evidence for "relay units," and so on (cf. Singer 1987), and therefore the claim of biological plausibility can be extended. Network elements such as unit types and connection patterns must be part of the discussion of neurobiological plausibility.

S&A refer to neurobiological findings that the temporal synchronous activity of cells in the cat's visual cortex supports the dynamic binding of visual features of objects. In other words, the temporal synchrony of neural firings supports pattern recognition and vision. What about explicit, reflexive reasoning? It is not at all obvious that the neurophysiological evidence for the dynamic binding of visual features carries over to multistep reasoning.

Conclusion. Biological plausibility, efficient reasoning, and the ability to make a number of predictions that allow psychological testing are the outstanding features of Shastri & Ajjanagadde's system. The absence of learning is disappointing and it is an important question how complex network structures can be generated efficiently with biologically plausible, connectionist methods. Apparently, there is work on the way on learning and an application of the system to natural language processing. This future work will show if some of the remaining problems can be solved.

ACKNOWLEDGMENT

I am grateful to Gerhard Brewka, Richard Coss, Georg Dorffner, Steffen Hölldobler, and Franz Kurfess for their comments on earlier drafts of this commentary.

Connectionism and syntactic binding of concepts

Georg Dorffner

Department of Medical Cybernetics and Artificial Intelligence, University of Vienna, A-1010 Vienna, Austria

Electronic mail: *georg@ai.univie.ac.at*

Shastri & Ajjanagadde (S&A) have done a remarkable job in modeling certain aspects of reflexive reasoning. However, some evaluation of their representations, as well as their solution to the binding problem, seems appropriate to put S&A's model in perspective with respect to its being "connectionist."

The motivation for S&A's introducing the idea of synchronous firings of nodes was the so-called binding problem. Even though this problem was pointed out early (see their references in sect. 2.1.1), it has really been brought into the foreground in the light of critiques of connectionist representations in neural networks (Fodor & Pylyshyn 1988b). According to these critiques, connectionist patterns of activation are merely sets of numbers lacking any structure. They therefore cannot naturally represent complex conceptual relations where different concepts have to be bound to their roles. In classical AI, due to the prevalent use of syntactic symbol structures, this was a non-issue. There, binding can easily be defined by assigning roles to syntactic position. S&A's response to the binding problem in connectionist networks is as remarkable as it is powerful, but it is also a very "classical" one. Defining binding through synchronous firings of nodes is identical to the syntactic solution in traditional AI systems. The remarkable thing about it is that the representation is moved into the temporal dimension. An example: Having *p-seller* and *Mary,* as well as *cs-obj* and *Book1* fire synchronously, respectively (example from sect. 3.1), is the same as representing this relation by using pairs of symbols in parentheses, meaning that the concepts in each pair are bound, for example ((*p-seller Mary*)(*cs-obj Book1*)).

The difference is that here only the spatial dimension is used for concatenating symbols, something that is obviously not possible in common neural networks (as units in a network cannot be repositioned). Now ((*p-seller Mary*)(*cs-obj Book1*)) does not seem to be the usual way of defining relations in the symbolic approach, as often a simpler form is used, such as *can-sell*(*Mary, Book1*). The binding in the latter representation is defined by implicitly assigning the two roles *p-seller* and *cs-obj* to the first and second position in the predicate, respectively. This may be possible in S&A's approach as well (such as defining that the first concept firing is the *p-seller*, and the second one the *cs-obj*), but it would make many things (such as mapping corresponding roles in different predicates) more difficult. In conclusion, we can say that S&A's solution to the binding problem is the *syntactic* solution of expressing concept relations, exploiting the temporal dimension in a clever and, as it turns out, even in a neurally plausible way.

Recent literature, however, has suggested that the syntactic solution need not be the only one (e.g., van Gelder 1990); it just happened to be the obvious one for symbolic models. Connectionist models – if taken in a much more general sense than in S&A's work – have the power to represent conceptual structures in a nonsyntactic, that is, superpositional way (Chalmers 1990; Pollack 1990; Sharkey 1992). The problem with the literature on this topic is that most such approaches are tested on problems equivalent to what the syntactic solution can achieve (e.g., the transformation of parse trees, as in Chalmers 1990). Thus I will briefly report results from our own work, which demonstrate that the superpositional way of representing compositional structure can lead to a much bigger step toward neurally and psychologically plausible models of reflexive reasoning.

The key to such a further step is what we call "soft compositionality." Connectionist literature on concept formation and distributed representations has shown that neural networks can implement "soft" concepts (Hinton 1986; Smolensky 1988) and rules (McMillan et al. 1991). Both can be viewed as fuzzy and analog entities which only in certain situations become discrete (e.g., in the process of unambiguous recognition of objects). S&A's model, on the level of single concepts and rules, could be viewed as a post-hoc approximation of such concept structures, explaining the situations where their discreteness counts. We want to argue that this cannot easily be said about bindings of concepts to their roles, as there should be a continuum between composite structures consisting of several concepts and their roles, on one hand, and holistic unstructured concepts, on the other.

Consider the following example of objects and their spatial relations to each other. Suppose I enter a room that contains, among other objects, a table with a chair on top. It is clear that I could represent this by a structure like *on-top*(*chair, table*) or ((*chair upper*)(*table lower*)). But when I have to reason reflexively, it will depend on the situation whether it really matters that there are two objects in a relation to each other. If I want to screw in a light bulb, I might deal with the whole thing as one concept (such as *ladder*). If my goal is not to bump into anything, it might be no concept at all, only a "fuzzy blob" causing some motor reaction. The hypothesis now is that in reflexive reasoning (exactly the kind S&A want to model) there can be a continuum between complex structures and holistic concepts in the representations used for reasoning. In other words, something might be represented best as a complex structure, as one whole, or as anything "in between." The important situations are the latter ones. I might have screwed in the light bulb and gone out of the room inclined to say that I dealt with one object in there. The question "Isn't there anything I can sit on?" however, can push my vague compositional representation of the two objects above threshold to permit the answer "yes, I guess I saw a chair on top of something else" (perhaps adding, "or was it underneath?"). In syntactic models such as S&A's, complex predicates are always complex predicates, and roles are always distinct. S&A do hint about a "soft" variant of rules (sect. 5.5), but "soft" compositionality in our sense would go beyond that, in that the existence of roles itself can be fuzzy and analog.

Dorffner and Rotter (1992) present a little model that, in a first step, partially achieves "soft" compositionality. It is based on the so-called binding vector (BV, Rotter & Dorffner 1990), which achieves superpositional representations similar to RAAM (Pollack 1990), but without prior learning. By starting with spatial relations and sensory input it also accounts for learning and grounding (another aspect where S&A have disappointingly little to say). The most interesting aspect with respect to S&A's work is that binding in the BV is also achieved by synchronous activation of concept and role. The difference is that after this, activations are sustained and superimposed onto each other and thus do not need to stay distinct.

In final conclusion, I want to argue that S&A's model of reflexive reasoning falls short of what other types of connectionist models could be capable of achieving. To be honest, research in self-organizing, distributed networks is still in its infancy and cannot directly compete with systems as complex as the ones S&A propose. Most of what has been said here must therefore remain a claim people working on those models will have to live up to. If they will, we could say that S&A's model, although powerful in many respects, is still much more symbolic than it is connectionist.

ACKNOWLEDGMENTS
I thank Erich Prem and Joachim Diederich for their comments on this text.

Dynamic bindings by real neurons: Arguments from physiology, neural network models and information theory

Reinhard Eckhorn
Department of Biophysics, Philipps-University, D-3550 Marburg, Germany
Electronic mail: eckhorn@pc1306.physik.uni-marburg.de

Does the S&A model use binding mechanisms similar to those of the brain? It is not clear from our experiences with the visual cortex whether the model of Shastri & Ajjanagadde (S&A) will still function as required if neural network dynamics are included that are similar to those of cortical neurons. This is called into question when we compare the rhythmic activities of the model and that of a real cortex. The most striking differences are the completely different dynamics (Eckhorn et al. 1988; 1990). Although the S&A model uses a simple phase-delay scheme for labeling and binding different entities and although it is driven by a rhythmic input, synchronized oscillations in the cortex are probably due to a self-organizing process among mutually coupled neurons; systematic, stimulus-specific phase delays of simultaneously occurring oscillatory events have not been observed to date. The relatively stable phase delays that have frequently been observed in about 10% of our recordings, however, did not occur in a stimulus-specific way, as would be required for bindings of entities in the S&A model. We can explain them as phase differences between signals from excitatory and inhibitory neurons that are locally coupled and involved in the generation of oscillations in the respective local assembly. In addition, oscillatory events (spindles) in the cortex are highly variable in their frequencies, amplitudes, durations, and delays in contrast to the signals in the S&A model.

The impressions from our observations of synchronization processes in the visual cortex and from our related neural network models suggest the following signal dynamics suitable for transient bindings of representations: The cortex may be able to represent a large number of entities nearly simultaneously by *forming synchronized oscillations of short durations and variable frequencies* in many different assemblies. Such oscillatory events are statistically independent in their signal courses as long as the entities they represent do not belong together. However, oscillatory active assemblies that are coupled by (mutual) connections can transiently form common oscillatory states of zero-phase difference by mutual synchronization and they may define by this process the transient binding between different entities. Such types of binding do not require distinct phases for distinct entities. Instead, entities may be defined by the internal coherence of the signals in a subassembly, and binding between subassemblies may be defined by the degree of transient signal correlation between signals of different subassemblies.

In addition to rhythmic synchronization, nonrhythmic synchronization might support dynamic binding. The binding process described above does not rely on rhythmic signals. Instead, nonrhythmic signals might introduce even higher degrees of freedom and thereby allow binding between more distinct entities (in the sense of the S&A model). This means that dynamic bindings might be achieved more generally by transient correlations between signals of any type, including oscillations as a reasonable case. In particular, irregular signals seem to be highly appropriate for the labeling of related entities, as has been shown in models of visual scene segmentation (Pabst et al. 1989). Participation of nonoscillatory signals in dynamic bindings is further supported by our observation that in the visual cortex oscillation spindles can be partially or completely suppressed by transient stimuli that drive cortical neurons of similar types synchronously in a stimulus-locked manner (Kruse et al. 1992). For these cases we have proposed that stimulus-locked synchronization serves for the transient binding (Eckhorn et al.

1990). This view is supported by everyday experience. Strong transient visual stimuli can be perceived rapidly, even in complex visual scenes, much faster than the cortex would require to generate several periods of a 40 or 50 Hz oscillation.

Neurons can process and transmit sufficient rates of information for the representation and routing of complex dynamic bindings. It is stated in S&A's introduction that neurons are slow computing devices and that they communicate relatively simple messages that can encode only a few bits of information (2 bits in 15 msec). This led S&A to the conclusion that a neuron's output cannot encode names, pointers, or complex structures that would be necessary, for example, to form dynamic representations and to propagate (route) them into specific directions. However, S&A may have underestimated the amount of transmitted information because they used a frequency code. Being aware of this, they added the argument that even if interspike delays are included in neural coding, the time available for a neuron to respond to its inputs is very limited, and so is the amount of transmitted information, because a presynaptic neuron can only communicate one or two spikes to a postsynaptic neuron before the latter must produce an output (see note 4).

These arguments do not convince us. First, a single cortical neuron generally has thousands of synapses at which one or two spikes can appear within a few milliseconds, leading to a complex time course of the postsynaptic potential with generally high information density. Second, signal processing in cortical neurons is not terminated by the generation of an output spike, because spikes generally do not "reset" the membrane potential, as can be seen directly in intracellular recordings from cortical neurons (e.g., Douglas et al. 1991). In addition, it has been shown rigorously that information rates in sensory and motor systems of mammals reach values of 300 bits/sec on a single nerve fiber if codes are chosen that match the signal transfer properties of the respective neurons (Eckhorn et al. 1976). This is equivalent to average information rates of 4.5 bits in 15 msec. However, if one calculates the actual time courses of information, rates of 6 bits in 15 msec often occur (Eckhorn & Poepel 1975). Such high rates in single neurons are assumed to be sufficient for the signaling of complex messages, including the routing of dynamic representations.

Much higher rates of information would be available at the "nodes" of the S&A model if the functional units of the nodes were realized by local groups of similar neurons. Such local ensembles might use probability coding on parallel output fibers, namely, the probability of discharge of any of these neurons would be the (quasi analog) signal that transmits the information of the ensemble. Information capacities in such systems using group codes can reach much higher values than those on single fibers because noise is reduced with increasing size of the group. If, for example, intrinsic neural noise is statistically independent (in the idealized case), then groups of N = 100 neurons can transmit information rates that are higher by a factor of 10 than those of a single neuron (proportional to the square root of N).

Although S&A's arguments for labeling and routing of complex representations by simply synchronizing the appropriate "nodes" are convincing for me, I would like to stress the point that at least in principle, other mechanisms might be used, because neurons have the capacity of signaling high rates of information during short intervals. In the S&A model the propagation time for a synchronized state to the next node requires about 20 msec, which is the cycle time observed in stimulus-induced oscillations (Eckhorn et al. 1988; Gray & Singer 1989). During 20 msec 8 bits can already be transmitted by a single real neuron, which is enough for signaling about 500 alternative states (each available for the communication of names or pointers).

Toward a unified behavioral and brain science

Jerome A. Feldman
International Computer Science Institute, 1947 Center St., Berkeley, CA 94704
Electronic mail: *jfeldman@icsi.berkeley.edu*

It is still commonplace to identify connectionist (or neural network) models with initially unstructured systems that are adapted through supervised or unsupervised learning. Shastri & Ajjanagadde's (S&A's) target article indicates how much richer the paradigm can be. What I find most remarkable is the progress that has been achieved in the first decade of the new wave of connectionist research. The central technical issue in this paper, connectionist variable binding, was viewed as intractable only a few years ago but now has a variety of competing solutions as outlined by S&A. And these solutions are no mere implementation of formal logic – the representations preserve many of the key computational advantages of connectionist models: parallelism, context-sensitivity, robustness, and evidential combination. A learning story needs to be added, but there is progress here also.

The most important thing about the target article is not the extent to which it is right or wrong, but the fact that we can now evaluate detailed models of complex phenomena that can lay claim to behavioral, biological, and computational adequacy. Not long ago this would have been impossible. It is obviously not easy to attack a hard problem (here, reflexive reasoning) from the various perspectives simultaneously, but we now have the tools for expressing such integrated models and this is beginning to have a profound effect on the unified behavioral and brain sciences.

Deconstruction of neural data yields biologically implausible periodic oscillations

Walter J. Freeman
Department of Molecular & Cell Biology, University of California at Berkeley, Berkeley, CA 94720
Electronic mail: *wfreeman@garnet.berkeley.edu*

Shastri & Ajjanagadde (S&A) provide a fine example of circular reasoning in their description of the "biological plausibility" of their model, in that the results to which they appeal constitute an AI-based interpretation of neurophysiological recordings rather than raw measurements of activity in the visual cortex of animals.

The "feature detector" interpretation deriving from the experimental work of Mountcastle (1957), Lettvin et al. (1959), Hubel and Wiesel (1962), and many others holds that when a complex sensory stimulus arrives in the sensory cortex, a small subset of neurons is vigorously excited and inhibited. Von der Malsburg and Schneider (1986) were the first to investigate systematically some of the ambiguities that arise when diverse stimuli can generate the same static neural response; they proposed a mechanism of phase-locked periodic oscillations to resolve them. The findings of Engel et al. (1990) and Eckhorn et al. (1988) appear to bear out his proposed solution, so that S&A feel justified in pointing to the similarity between putative visual cortical function and their model based on periodic orbits and phase-locked pulses at some common frequency in their net.

It is in the aspect of periodicity that their model falls short of the biological data; the fault, however, lies not with S&A but with the interpretation by the biologists. If cortical neurons were routinely observed to fire periodically at a designated network frequency then the von der Malsburg interpretation would be amply justified. Periodically firing neurons are indeed found on occasion, particularly in the cat, which for unknown reasons, has a peculiar tendency to narrow band oscillations in all of its sensory cortices; but these neurons form a small tail in a distribution of firing rates and patterns and the great majority of neurons yield pulse interval histograms that conform more to the Poisson than to the periodic distribution. Also the time-lagged covariances between the pulse trains of pairs of neurons tend to be vanishingly small (Abeles 1991), which would not be so if the neurons usually shared a firing frequency, whether or not they were in phase.

A further problem is that the mean firing rates of most cortical neurons are considerably less than the prevailing peak frequencies of cortical dendritic potentials (local field potentials or electroencephalograms [EEGs]) in the gamma band (including the 40 Hz). This fact is obscured by such techniques as multiple unit extracellular recording, which is a form of spatial ensemble averaging over a local cortical domain; correlation analysis of spike trains, which is a form of time ensemble averaging that enhances the appearance of narrow band oscillation by expressing the time-variance of a frequency as the decaying envelope of the correlation oscillation at the center frequency; and spike-triggered averaging of EEGs, which invokes the spatial ensemble averaging that is inherent in dendritic extracellular field potentials (Freeman 1975; 1991) and the time ensemble averaging that enhances the center frequency of a distribution of frequencies. Again, the cat (from which the bulk of new results in this development have been taken) yields particularly simple wave forms, but unaveraged records from the lagomorph and simian visual cortices reveal broad spectrum EEG activity relating to goal-directed behavior on single trials (Freeman & van Dijk 1987), which is oscillatory, to be sure, but strongly aperiodic.

In brief, pulse trains and EEG waves are mostly aperiodic. It is the requirement of AI-based modeling that leads to manipulation of the data for the extraction of center frequencies and to the suggestion that there is rapid convergence of visual cortical dynamics to limit cycle attractors. Now definitions of "phase locking" and "phase coherence" (as distinct from spatial coherence of broad spectrum activity) can only be based on the existence of discrete frequencies. The characteristically sloppy wave forms seen in raw data indicate that the cortex is rather indifferent to precise control of the frequencies of its pulse trains and dendritic current amplitudes, and that it allows them to vary continually seemingly at random. But the phase of a continuous frequency distribution cannot be defined for these events.

Even with the techniques of data refinement noted above, which suggest that narrow band oscillations are capable of coming into synchrony in time periods as short as one cycle (20 to 40 msec), there is a reported spread of coupling of ± 27° to ± 54° at 25 to 50 Hz and a 95% confidence range of 108° to 216°. If these confidence intervals hold in the presented model, the short time segments of 0.1 sec for the perceptual frames that are invoked by S&A will not yield adequate reliability for readout by detectors of phase lockings from a transmitting array. S&A note some of the further unresolved difficulties regarding the management of multiple-phase modes and the compounding of the difficulties when "soft" rules are brought into play, by which continuous gradations of the degrees of synchronization are used. Hence S&A's model is biologically implausible.

The hypothesis underlying S&A's formulation of the "binding problem" is that the visual cortex operates by extracting strong correlations among a small subset of very active neurons in any given time segment. An alternative hypothesis is that the cortex operates by extracting weak covariances among very large populations of neurons whatever the magnitudes of their individual activity. On this premise a nonlinear dynamics can be constructed (Freeman 1991) that envisions the existence of multiple chaotic states and "itinerant" trajectories among them (Tsuda 1991). Here spatial coherence is crucial, and although it can occasionally be detected as phase locking through massaging of the data, the instantaneous frequency can and does vary pseudorandomly over the short term. Simulations suggest that chaotic dynamics may be unusually powerful at solving pattern recognition tasks (Yao et al. 1991).

Biological memory systems are well known for their vagaries (Bartlett 1934), and a case has been made that the essential neural dynamics underlying the construction (not reconstruction) of images during acts of remembering is chaotic (Freeman 1991; Skarda & Freeman 1987). S&A note properly that they are not intending to elaborate a model for brain function so their appeal to "biological plausibility" seems inappropriate. Their model for a commonsense knowledge base may be fruitful in managing retrieval of information from large libraries, but only if the "knowledge" has already been once removed from the real world, just as the purported "limit cycle" behaviors of nerve cells have already been deconstructed from the actual performance of neurons in living brains.

Must we solve the binding problem in neural hardware?

James W. Garson
Department of Philosophy, University of Houston, Houston TX 77204-3785
Electronic mail: *phil0@jetson.uh.edu*

Shastri & Ajjanagadde's (S&A's) idea of representing variable binding with signal synchrony and implementing deduction by entrainment of signals is attractive. However, S&A postulate a very specialized and rigid neural architecture to accomplish these tasks. S&A's model verifies that a net can implement fast (but limited) reasoning by exploiting signal synchrony. That does not tell us much about the brain, however. It would be a miracle if the brain contained networks involving collectors, enablers, rho-btu nodes, tau-nodes, concept clusters, and switches exactly as S&A describe. Our understanding of the task to be modeled and the machinery available comes nowhere near to constraining the solution this tightly.

S&A's project belongs to a paradigm in connectionist research that attempts literal implementation of machinery (such as binding) drawn from classical AI. There is an alternative connectionist paradigm that takes the project of understanding binding much less literally (Elman 1991; Servan-Schreiber et al. 1989). It postulates a very simple recurrent net architecture and successfully trains nets (with modified backpropagation) on tasks that are traditionally thought to involve variable binding. In this empirically minded paradigm, no attempt is made to define ahead of time the subtasks or their manner of implementation. That is to be discovered by the net, not stipulated. Here solutions to the "binding problem" emerge from weight selection in a general purpose architecture that uses *distributed* rather than local representations. This line of research has not tackled reasoning directly, but it shows at least that some implicit binding can be handled without special architecture.

The strong point of S&A's proposal is to display the advantages of exploiting time in connectionist representation. However, the same strategy is also exploited by nets trained in the empirical paradigm. Here syntactic structure is represented by the system's *trajectory* through phase space (van Gelder 1991). Task classicists characterize as involving binding are accomplished by setting weights so that these trajectories are constrained in the right way. Empirically constructed systems do not represent arguments and fillers literally with nodes or groups of nodes. Representation is distributed, and it is only by principle component analysis of pattern sequences on hidden units that the outlines of the classical argument-filler ideas can be brought into focus. This makes it hard to understand processing with distributed representations in classical terms. However, distributed representations are more difficult for a net to acquire and manipulate (Chalmers 1990).

Empirical model building is no better than classical at proving how the brain actually functions. However, it does sensitize us to the idea that understanding reasoning in the brain may require transforming classical constructs rather than models for their literal implementation. Given the possibility that distributed coding avoids the need for specialized architecture, speculation on details of an architecture specifically designed to support reasoning on local (or semilocal) representations is premature.

I have a number of more detailed worries about S&A's proposal:

1. I wonder how the theory deals with negation. How do we handle negative conclusions, and (say) rules of the form: No *A* is *B*? It is true that negation can be implicitly expressed in production rules by replacing a negative consequent by the corresponding positive antecedent. But applying this strategy to "No *A* is *B*" leaves the consequent empty, and there is no provision for empty consequents in S&A's scheme. As a useful benchmark for how serious problems involving negation might be, I invite S&A to explain how "No one is taller than himself" could be a reflexively reached conclusion.

2. I still wonder how rules can be learned. The strategy (described in sect. 10.6) of tuning weights to establish the right connections between predicates only works if generic links between all the right predicates are already available. Providing links for all possible connections between all possible predicates sets off a combinatorial explosion. (To make matters worse, there have to be separate links for forward and backward reasoning since neurons do not conduct in two directions.) So S&A's model predicts that many rules simply cannot be learned because the predicates happen not to be "neighbors." Distributed representation avoids this problem because "links" between arbitrary concepts are forged as processes rather than in physical space.

3. This problem is compounded when we turn to propositional attitudes, prepositions, and other modifiers. Consider all the sentences we can construct by adding and deleting elements to "Al saw Carol deceptively sell Dog1 to Ed in the presence of Frank under the influence of alcohol in a park." To represent these sentences, modifiers and propositional attitudes must be dealt with as arguments of the main verb ("sold" in this case). (We certainly cannot represent every possible combination as a separate predicate.) But then each predicate must have arguments for all conceivable (and eventually learnable) modifiers and propositional attitudes. The space investment in each predicate is massive. Furthermore, I see no practical way to account for reflexive reasoning from "Al sees that *p*" to "Al knows that *p*." (Reflexive reasoning for iterated attitudes or sentences where modifier scope matters would also be impossible.) The argument structure of English is too complex and open-ended to be written into our neurons.

S&A may complain that I take their model too literally. They say in section 1.4 that their model is not intended as a blueprint of how the brain performs. However, if their model is not at least a provisional neural wiring diagram, then it is not clear how the arguments involving neural time and space constraints they cite are relevant in supporting their model over its competitors. If links indicate functional structure, we must wait until we know how links are implemented before we apply considerations concerning (say) speed of neural conduction.

Self-organizing neural models of categorization, inference and synchrony

Stephen Grossberg

Center for Adaptive Systems and Department of Cognitive and Neural Systems, Boston University, Boston, MA 02215

Electronic mail: *cindy@cns.bu.edu*

Shastri & Ajjanagadde (S&A) note how carefully controlled synchrony may subserve a rapidly evolving inference, but they do not show how the knowledge being synchronized can be learned, and how this learning process leads to the desired synchrony relationships. A recently discovered family of supervised learning, categorization, and prediction architectures provides insight into aspects of this fundamental problem. These neural architectures are generically called ARTMAP (Carpenter & Grossberg 1991; 1992; Carpenter et al. 1991; 1992).

ARTMAPs can learn arbitrary analog or binary mappings between learned categories of one feature space (e.g., visual features) to learned categories of another feature space (e.g., auditory features). They perform well in benchmark studies against alternative machine learning, genetic algorithm, or neural network models. This may be because the Adaptive Resonance Theory modules that go into ARTMAPs were derived from a study of brain data (Grossberg 1987; 1988). In particular, ARTMAPs can autonomously learn, categorize, and make predictions about:

1. Rare events: A successful autonomous agent must be able to learn about rare events that have important consequences even if those rare events are similar to a surrounding cloud of frequent events that have different consequences. *Fast learning* is needed to pick up a rare event on the fly.

2. Large nonstationary data bases: Rare events typically occur in a nonstationary environment whose event statistics may change rapidly and unexpectedly through time. ARTMAP contains a self-stabilizing memory that permits accumulating knowledge to be stored reliably in response to arbitrarily many events in a nonstationary environment under incremental learning conditions until the algorithm's full memory capacity, which can be chosen arbitrarily large, is exhausted.

3. Morphologically variable types of events: In many environments, some information, including rulelike inferences, is coarsely defined, whereas other information is precisely characterized. ARTMAP is able to adjust its scale of generalization automatically to match the morphological variability of the data. It embodies a Minimax Learning Rule that jointly minimizes predictive error and maximizes generalization using only information that is locally available under incremental learning conditions in a nonstationary environment.

4. Many-to-one and one-to-many relationships: Many-to-one learning takes two forms: categorization and naming. For example, during the categorization of printed letter fonts, many similar exemplars of the same printed letter may establish a single recognition category. All categories that represent the same letter may be associatively mapped into the letter name or prediction. This is a second many-to-one map arising for cultural, not visual, reasons.

One-to-many learning is used to build up expert knowledge about an object or event. A single visual image of a particular animal may, for example, lead to learning that predicts animal, dog, beagle, and my dog "Rover." In many learning algorithms the attempt to learn more than one prediction about an event leads to unselective forgetting of previously learned predictions for the same reason that these algorithms become unstable in response to nonstationary data.

ARTMAP systems exhibit properties 1-4 because they implement a set of heuristics qualitatively different from those of error-based learning systems:

5. Pay attention: An ARTMAP can learn top-down expectations (also called prototypes, primes, or queries) that can bias

the system to ignore masses of irrelevant data. These queries "test the hypothesis" that is embodied by the category as they suppress features not in the prototypical attentional focus.

6. Hypothesis testing and match-based learning: The system actively searches for recognition categories, or hypotheses, whose top-down expectations provide an acceptable match to bottom-up data. The top-down expectation focuses attention upon, and binds, that cluster of input features that it deems to be relevant.

7. Choose globally best answer: After learning self-stabilizes, every input directly selects the globally best matching category without any search.

8. Calibrate confidence: A confidence measure, called *vigilance*, calibrates how well an exemplar matches the prototype that it selects. If vigilance is low, even poorly matching exemplars can then be incorporated into one category, hence compression and generalization are high. If vigilance is high, few exemplars activate the same category, hence compression and generalization are low. In the limit of very high vigilance, prototype learning reduces to exemplar learning. The Minimax Learning Rule adjusts the vigilance parameter just enough to initiate hypothesis testing to discover a better category, or hypothesis, with which to match the data. In this way, a minimum amount of generalization is sacrificed to correct the error.

9. Rule extraction: At any stage of learning, a user can translate the state of an ARTMAP into an algorithmic set of if-then rules. ARTMAPs are thus a new type of self-organizing production system. The Minimax Learning Rule determines how abstract these rules will become.

10. Properties scale: All the desirable properties of ARTMAPs scale to arbitrarily large problems.

11. Working memory: The input level of an ARTMAP may be a working memory designed so that any grouping of its stored events can be stably learned in real time. Using STORE working memory models (Bradski et al. 1992a; 1992b), temporally evolving rules may be learned.

12. Temporal synchrony: The first ART articles (Grossberg 1976; 1978) predicted that visual cortical codes could be expressed by synchronous oscillations in which cooperatively linked cells oscillate in phase and that oscillations could be replaced by equilibrium points if no "slow" variables, such as inhibitory interneurons or chemical modulators, exist. Within ART, a synchronized oscillation can occur when bottom-up feature-selective and top-down expectation signals fuse into an attentive resonance that can support new learning and a conscious perceptual experience. The predicted linkage between standing waves, attention, learning, and conscious experience has recently attracted much interest (e.g., Crick & Koch 1990b).

After ART was introduced to analyze data about attentive learning and recognition, Grossberg and Mingolla (1985a; 1985b) modeled processes of preattentive vision. A new type of bipole cell was predicted to link perceptual features cooperatively into emergent boundary segmentations within a Boundary Contour System (BCS). Grossberg and Somers (1991; 1992) have demonstrated that both the BCS and ART circuits can link cells cooperatively into rapidly synchronizing oscillations over large cellular distances. The oscillation is not necessary for binding per se, because features can be bound in a way that explains large data bases using BCS and ART models in which no oscillations occur. The oscillations provide an extra degree of freedom that scales the amount of asynchrony that can be tolerated before the wrong object parts may be bound together.

ACKNOWLEDGMENTS
The author wishes to thank Kelly Dumont, Diana Meyers, and Robin Locke for their valuable assistance in the preparation of the manuscript. The author is supported in part by the Air Force Office of Scientific Research (F49620-92-J-0225), DARPA (AFOSR 90-0083), and the Office of Naval Research (ONR N00014-91-J-4100).

Competing, or perhaps complementary, approaches to the dynamic-binding problem, with similar capacity limitations

Graeme S. Halford
Psychology Department, University of Queensland, Queensland 4072, Australia
Electronic mail: *gsh@psych.psy.uq.oz.au*

Shastri & Ajjanagadde (S&A) have provided an impressive demonstration of the power of synchronous activation to handle the dynamic-binding problem in reflexive reasoning; however, predicate-argument bindings can also be handled using the tensor product approach advocated by Smolensky (1990). Halford et al. (1993) have proposed an analogical reasoning model in which an N-place predicate is represented by a tensor product of rank $N + 1$, with one vector representing the predicate and N vectors representing arguments. We therefore have a situation in which two very different approaches have similar achievements. Synchronous activation can handle reflexive reasoning and analogical reasoning (Hummel et al., in press); tensor product representations handle production systems (Dolan & Smolensky 1989) and memory retrieval (Humphreys et al. 1989), as well as analogical reasoning.

These approaches may be competitive, or they may be complementary, so that synchronous activation models deal with reflexive or implicit reasoning and tensor product models might be more appropriate for reflective or explicit reasoning. Tensor product representations can represent certain properties of relations that do not appear to be possible for the synchronous activation approach. A relation $R(a,b, \ldots ,n)$ can be handled by a tensor product of rank $N + 1$ (Halford et al. 1993). This not only represents the predicate-argument bindings but also the interactions within the structure. For example, the tensor product representation of $R(a,b,c)$ represents the influence of c on $R(a,b)$, the influence of b on $R(a,c)$, and the influence of a on $R(b,c)$. The synchronous activation approach can handle slot-filler bindings but it does not appear able to represent these higher-order relations that are important to complex concepts.

It is possible to collapse over any vector in the tensor product (Humphreys et al. 1989), so any subset of the possible relations can be represented. For example, given that $R(a,b,c)$ is represented by a rank-4 tensor product, $R(a,b)$, $R(b,c)$, $R(a,c)$, and $R(a,b,c)$ can be processed. Furthermore, any argument can be retrieved given the predicate and remaining arguments, and the predicate can be retrieved, given the arguments. Thus, the tensor product representation has the flexibility and power that are characteristic of explicit or reflective reasoning.

There are interesting correspondences in the way capacity limitations are handled by the two models. In synchronous activation models the number of distinct phases is limited to between 5 and 10, whereas Halford et al. (1993) propose, after a review of the working memory literature, that tensor product models are limited to rank 5. They argue that the flexibility and power of tensor product models in handling complex reasoning requires each argument to be represented by a separate vector, which has the status of a dimension in that it provides an independent source of variation. This corresponds to a distinct entity in S&A's terms. The information represented by each dimension, or each distinct entity, is variable, often over a wide range, but the number of dimensions, or entities, is limited by the rank of the tensor product, or number of phases. Thus the models agree that the limit is not in the amount of information expressed by each entity or dimension but in the number of independent dimensions (entities) represented in parallel.

It therefore appears that synchronous activation and tensor product models have converged on a theoretical basis for the concept of a chunk (Miller 1956), which is an independent item of information of arbitrary size and is the unit that best defines human-processing limitations. Both explain the finding that the amount of information that can be represented in any one item (the chunk size) is variable over a wide range but the number of independent items (number of chunks) that can be represented in parallel is very restricted.

ACKNOWLEDGMENT
This work was supported by a grant from the Australian Research Grants Scheme.

Rule acquisition and variable binding: Two sides of the same coin

P. J. Hampson
Department of Applied Psychology, University College, Cork, Ireland
Electronic mail: *stay8026@iruccvax.ucc.ie*

Shastri & Ajjanagadde (S&A) provide a number of important contributions to our understanding of the psychology of reasoning and inference and its modeling with connectionism. Their distinction between reflexive and reflective reasoning is well made and in accordance with a growing consensus that many types of human inference, particularly those directly involved in language comprehension, are simply too fast for the sort of deliberative processing associated with, say, solving syllogisms. The network models discussed have the advantages of being computationally simple and neurologically plausible and make good contact with other areas of research such as working memory. On the whole the paper makes a serious and distinguished contribution to the area and will, most likely, be widely cited.

Some clarification of the manner in which rules are acquired would be in order, however, and would allow the theory to develop further. S&A are obviously alert to this and touch on a crucial point when they state that they are considering learning "in the context of preexisting predicates and concepts where it is desired that the cooccurrence of events should lead to the formation of appropriate connections between predicate arguments" (sect. 10.6). Recent developments in the learning theory on stimulus equivalence and relational frame effects bear on this issue and indicate the importance of quite extensive exposure to such cooccurrent events. Briefly, these developments suggest that large amounts of bidirectional training across a number of domains are required by children before even the simplest stimulus equivalence relation can be acquired. For example, even simple symmetric relations between entities are typically not exhibited by children of less than two years old. It seems rather that a large number of forward (*A* goes with *B*) and backward (*B* goes with *A*) mappings must be experienced across a series of domains before domain invariant information (rules) can emerge. Once these are acquired, merely learning that *C* goes with *D* is sufficient for the reverse relation, *D* goes with *C*, to be inferred (see Hayes [1991] for a detailed account of relational frame theory). The key point is that rules themselves are never acquired directly or within one domain; instead, the invariant information that instantiates them emerges throughout a series of behavioural interactions across several domains.

Viewed thus, rule acquisition is the reverse of the variable-binding problem. Variable binding entails the attachment of content with a rule (or an abstract structure) for use in a particular situation; rule acquisition and instantiation involves the functional detachment of common structure from a set of variable contents for use in future situations with new content. In S&A's terms this means that examples of both forward and backward pairings between predicate arguments must be explicitly experienced, in a variety of situations, before the sorts of inferences they discuss are possible. In our own work we have modeled some of these effects and shown how performance on inference tasks, including reasoning about kinship relations, improves as a function of exposure to related domains (Barnes & Hampson 1992). A solution to the acquisition problem in the context of this model would probably help solve the problem of memorizing facts or of converting dynamic bindings to the static patterns S&A also identify.

It would also be interesting to see how easily the model extends to analogical thinking. Both analogical thinking and the more typical inferences considered by S&A can be construed as similar processes in that both involve a match between domain invariant information. It should not be impossible to extract higher-order relations between predicate arguments across many pattern sets and to use these to support analogical reasoning in a system such as S&A's.

Finally, an area that the field as a whole could now usefully consider is the movement from reflective to reflexive reasoning, which might be expected to follow practice in certain situations. This mode-shift in reasoning seems likely given the assumption that reflective reasoning involves conscious deliberation and reflexive reasoning is more automatic, and the evidence that practice generally shifts processing in the direction of automaticity. Previous accounts of strategy shifts in reasoning and problem solving have focused on changes in the representation used, such as from visual image to verbal (e.g., Kosslyn et al. 1977); it might be more fruitful now to consider whether (in addition or as an alternative to these representational shifts) there is a shift from reflective to reflexive reasoning modes – though as S&A astutely point out, there may well be some situations that simply do not permit reflexive reasoning (sect. 8.2.5).

Not all reflexive reasoning is deductive

Graeme Hirst[a] and Dekai Wu[b]
[a]Department of Computer Science, University of Toronto, Toronto, Ontario, Canada M5S 1A4; [b]Department of Computer Science, University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong
Electronic mail: *[a]gh@cs.toronto.edu; [b]dekai@uxmail.ust.hk*

Shastri & Ajjanagadde's (S&A's) model is a well fleshed-out proposal linking conceptual inference with neural representation. Assigning one phase to each concept occurrence is a clever idea that is worthy of further development. In this commentary, we discuss some of the problems that remain.

In their note 3, S&A see their notion of reflexive reasoning as a generalization of the well-established notion of automatic processing. In fact, the converse would seem to be true; S&A talk about rapid deductive inference as if it were the only kind of reflexive, or automatic (unconscious? – cf. Velmans 1991), reasoning people perform. In reality, it is just one of many kinds, some of which are quite general and others of which are very specific.

1. Determining a *probable relationship* between two or more concepts. This is the kind of reasoning we do when we interpret novel (unlexicalized) nominal compounds such as *temporal pattern matcher*. (Downing [1977] has shown that the class of relationships between elements in nominal compounds is large and unconstrained; but see Levi 1978.)

2. Computing the *semantic distance* between two concepts, which is a fundamental part of such automatic reasoning as lexical disambiguation (Charniak 1983; Hirst 1987; 1988; Hirst & Charniak 1982) and certain kinds of problem solving (Hendler 1987). This kind of reasoning was achieved in the work cited by means of marker passing; but, notwithstanding S&A's remarks about the similarity of their approach to marker passing, the computation of semantic distance does not seem amenable to any kind of phase encoding, for it relies crucially upon a *static* property of the knowledge base – that the physical distance between representations of concepts corresponds reasonably well to the semantic distance.

3. *Elaborative inferences* such as supplying typical values for roles whose fillers are left implicit or unspecified. For example, subjects reading *Mary stirred the coffee* showed subsequent facilitation for *spoon* in a word-completion task (Whitney & Williams-Whitney 1990). (We speculate that this is not mere word association; for example, *Mary stirred the paint* would facilitate *stick* but not *spoon*. This hypothesis is presently being tested in work in progress by the first author in collaboration with Michael K. Tanenhaus and Gail Mauner.) Similarly, Cahill and Mitchell (1987) found that reading a passage that described a goal and a precondition for achieving it led to the inference of a plan. The exact conditions under which elaborative inferences are made has been the subject of some debate in the literature (Dosher & Corbett 1982; Lucas et al. 1990; Whitney & Williams-Whitney 1990); here we need only note that they do occur under at least some conditions.

4. The *interpretation* of direct and indirect speech acts and of discourse repairs and the recognition, in general, of intent, as distinct from literal meaning, in discourse. Such tasks, when described in full logical detail, are extraordinarily complex (cf. Allen & Perrault 1980; Cohen et al. 1990; McRoy 1993; McRoy & Hirst 1993). Although such interpretation is surely based on compiled rules rather than carried out from first principles each time (cf. Gibbs 1983), it remains automatic and not deductive. More generally, much expert reasoning is reflexive interpretation, involving the recognition and categorization of patterns in the domain of expertise (see, e.g., Cooke 1992, and the references cited therein). (Note that this is *not* categorization in the sense that S&A use that word in their sect. 2.4.)

5. *Abductive inference,* which can also be extremely rapid. On this, S&A allude to another paper of theirs (Ajjanagadde 1991), but offer no details.

Moreover, S&A are not clear enough about neural plausibility – specifically, whether individual neurons or ensembles in their representation can possibly have biological correlates. On the one hand, the imposition of detailed constraints on connectivity and firing rate implies a biological interpretation. On the other hand, their model is essentially localist; the representation seems biologically implausible even when symbolic neurons are replaced by localist ensembles late in the development (sect. 7.3). A symbolic representation is perfectly acceptable at an abstract level of explanation, but experimentation with timing parameters makes sense only if the representation itself is neurobiologically consistent.

We believe that in certain ways the model is closer to marker passing than the authors suggest. They refer to Fahlman's (1979) original proposal (sect. 3) and to generate-and-filter systems that "evaluate the relevance of . . . paths [after collisions]" (note 14) (e.g., Charniak 1983; Hendler 1987; Hirst 1987; Norvig 1989). However, other marker-passing models have been proposed where collisions generate inferences in first-come, first-served fashion (e.g., Martin & Riesbeck 1986; Wu 1989). Markers carry variable-binding information rather than what S&A call "backpointers to the original and immediate source of the marker" (ibid.), and markers arriving at the same node are considered to "collide" only if their bindings match. If we further impose a phase for each variable, a very similar model results.

The model as proposed does not accommodate uncertainty. Indeed, the "more complex messages" (ibid.) that are carried by markers are also partly to handle probabilities (Wu 1989). S&A suggest integrating temporal synchrony with an earlier evidential system (sect. 9.2), but defining a probability distribution over inferences is not straightforward when variable bindings are permitted. This is because the binding hypotheses themselves interact, both logically (say, by mutual exclusivity) and statistically. So it is not even clear what the functional specification ought to be.

An approach that might therefore help is the maximum-entropy distribution as generalized for hypotheses involving arbitrary variable bindings (formulated by Wu 1992a; 1992b). Because variable bindings make full probability computation too expensive, Wu also gives a robust approximate method, AME (approximate maximum entropy), that allows arbitrary subpartitions of probabilistic constraints and hypotheses to be preselected. How tractable approximations such as this could be incorporated in a fixed connectionist architecture is an important issue for future research.

S&A compare their temporal-synchrony method for binding with the reduced-description approach (sect. 9.4). However, the comparison is based on the encoding of rules as directed dependency graphs. Combining reduced descriptions with rule graphs is inappropriate because the object of reduced descriptions is to avoid representing rules locally (e.g., Pollack 1988; 1990; Stolcke & Wu 1992). Also, contrary to S&A's statement that reduced-description approaches "will also have to be augmented in order to deal with noise," inherent resistance to noise is one of the nice properties that results from distributed representation.

ACKNOWLEDGMENTS
This commentary was written while the second author was at the Department of Computer Science, University of Toronto. Preparation of the commentary was supported by a grant from the Natural Sciences and Engineering Research Council of Canada.

On the artificial intelligence paradox

Steffen Hölldobler
Intellektik, Informatik, Technische Hochschule Darmstadt, D-6100 Darmstadt, Germany
Electronic mail: *steffen@intellektik.informatik.th-darmstadt.de*

Shastri & Ajjanagadde (S&A) claim that their "computational model takes a step toward . . . resolving the artificial intelligence paradox," namely, the gap between the ability of humans to draw a variety of inferences effortlessly, spontaneously, and with remarkable efficiency on the one hand and the results about the complexity of reasoning reported by researchers in artificial intelligence on the other hand. This claim seems to be too strong. S&A's logic has certain special features. These features are quite remarkable and are the result of an attempt to find a class of formulae which is as expressive as possible and whose satisfiability can be decided by the propagation of rhythmic activity in parallel time bound by the length of the shortest proof and with space bound by the size of the formula. Nevertheless, from a logic point of view the expressive power of S&A's system is fairly limited. And the mere fact that artificial intelligence researchers have not investigated this particular logic does not imply that a significant step toward resolving the artificial intelligence paradox has been made.

But have artificial intelligence researchers really not investigated S&A's logic? Because of the imposed restrictions, S&A's system need not unify expressions but the matching operation suffices. Whereas unification is inherently sequential (Dwork et al. 1984), matching is known to be parallelizable in an optimal way (Ramesh et al. 1989). There is also a striking similarity between S&A's reasoning mechanism and certain reduction techniques applied in automated theorem provers such as the evaluation of isolated connections (Bibel 1988). For example, if each variable occurring in the conditions of a rule occurs also in the conclusion of a rule then a query, all of whose arguments are bound to constants, can be solved by evaluating isolated connections only in precisely the same way that S&A's system solves this query. As shown by Hölldobler (1990) the evaluation of isolated connections can be applied in parallel. Moreover, if a formula is as restricted as mentioned above and can be solved by applying this reduction technique only, then the bounds on time and space are comparable to the bounds in S&A's system. But whereas the reduction techniques in an automated theorem prover are applied in the larger context of proving the satisfiability of an unrestricted first-order formula, S&A's system is designed to show the satisfiability of a very special class of formulae and, hence, is more elaborate for this special class. If the similarity between reduction techniques applied in automated theorem provers and the computational model presented in this article holds for most of the special features, then S&A's work shows that automated theorem provers which apply these reduction techniques in parallel are adequate in the sense that they solve simpler problems faster than more difficult ones. Unfortunately, the authors have not investigated this similarity.

The results of the target article would be a step toward resolving the artificial intelligence paradox if commonsense reasoning problems were expressible in S&A's logic. The paper contains some predications on this topic and it remains to be seen whether these predictions hold. If they do then the gap between the ability of humans to draw a variety of inferences as if it were a reflex and the results about the complexity of reasoning reported by researchers in artificial intelligence is not a paradox at all. If problems that can be solved effortlessly by humans can be expressed in S&A's logic, then these problems are just simpler than the problems investigated in the artificial intelligence community.

ACKNOWLEDGMENT
I would like to thank Antje Beringer, Joachim Diederich, and Franz Kurfess for their comments on this commentary.

Distributing structure over time

John E. Hummel[a] and Keith J. Holyoak[b]
Department of Psychology, University of California at Los Angeles, Los Angeles, CA 90024
Electronic mail: [a]jhummel@cognet.ucla.edu; [b]holyoak@cognet.ucla.edu

Shastri & Ajjanagadde (S&A) have made an important contribution to the development of a connectionist representational theory that accounts well for the fundamental systematicity of human reasoning. The most basic contribution of their work is its demonstration that a connectionist-style model can represent and use propositions and, more generally, structured information. Despite the current flurry of interest in synchrony for binding within the neural network community, comparatively few modelers have proposed serious accounts of how synchrony can actually perform useful work. Typically, networks are shown to establish synchrony and the functional significance and capacity of that synchrony is left to the imagination. In contrast, S&A provide an explicit account of the representation of structure via synchrony in a connectionist-style architecture.

Further work on the use of synchrony in knowledge representation is needed, and a number of important issues deserve careful scrutiny. We consider one of the most basic issues: the inherent tradeoff between distributed representations and systematic bindings among units of knowledge. The primary advantage of a distributed representation is its ability to capture naturally the similarity structure of the represented domain (similar entities can share a greater number of units in the representation than dissimilar entities). The disadvantage is that binding systematicity decreases (i.e., the likelihood of a binding error increases) with the extent of distribution. Consider the extreme cases. In a purely localist representation, no binding errors are possible. If there are N units, each representing a different concept, then the network can simultaneously represent its entire vocabulary of concepts without any ambiguity about what is being represented. The other extreme is the completely distributed case, in which each of the 2^N binary patterns possible over N units represents a distinct concept. In this case, no two patterns may be superimposed without spuriously creating a new pattern; in the event of superposition, binding errors are inevitable. Intermediate degrees of distribution present intermediate likelihoods of binding errors.

The value of synchrony is that it allows a network to use a distributed representation without being subject to binding errors, thereby alleviating the tradeoff between similarity and systematicity. There is a catch, however, which we term the *one-level restriction:* Synchrony can only represent element bindings at one level of abstraction or hierarchy at a time. That is, synchrony cannot simultaneously represent the binding of elements to each other and also the bindings of the units within the patterns representing those elements. This restriction is evident in S&A's model. The representation of propositions is distributed over multiple predicate and object units but the predicates and objects themselves are strictly localist. The one-level restriction implies that hierarchical structures will be difficult to represent. It is unclear, for example, how S&A would extend their system to represent propositions such as "Jane knows that Ted gave Mary flowers," in which an entire proposition (rather than a simple object) is bound to the role of "what is known."

The one-level restriction has other important implications for S&A's model. A basic strength of the model is its capacity to stack an unlimited number of predicates on top of an object without additional cost (i.e., any number of predicate units may fire on a given time slice). This capacity is critical both to the model's operation (it is directly responsible for its ability to "search" in parallel down multiple inference paths) and for its behavioral predictions (specifically, that many predicates modifying few objects should require less capacity than few predicates modifying many objects). But S&A's model can only stack predicates because its representations of predicates are nonoverlapping (localist). If S&A adopted a distributed representation for predicates then stacking would entail sacrificing systematicity of bindings. S&A's use of localist predicates is thus more than a notational convenience; it is an integral part of the model's architecture with far-reaching implications.

The one-level restriction does not imply that it is impossible to use a distributed representation at more than one level of abstraction; rather, it implies that if the lower-level (e.g., predicate) representation is distributed then multiple elements of this kind cannot in general be combined within a single time slice. That is, S&A could represent their predicates in a distributed fashion but they would no longer be able to stack them. In our own work (Hummel & Holyoak 1992; Hummel et al., in press) we have explored the use of synchrony to represent propositions. Like S&A's model, ours uses synchrony to bind objects to case roles within propositions; but unlike S&A's model, ours uses a distributed representation of objects and predicates. The benefits of our representation are all those typically associated with distribution (e.g., similarity, automatic generalization, etc.). The cost is that our model cannot stack predicates in the same unbounded manner as S&A's model. Rather, it represents the binding of one object to only *one* case role per time slice.

We have come full circle, returning to the tradeoff that originally motivated the use of synchrony. S&A's model and ours represent opposite extremes of this tradeoff, only this time, synchrony – already assumed – is not available to ease our dilemma. Some degree of distribution at the level of predicates seems necessary; thus S&A are forced to search an *IS-A* hierarchy to capture similarity relations. And our restriction of one object-to-case-role binding per time slice may entail processing that is too serial for the type of reflexive reasoning performed by S&A's model. An interesting question concerns what compromises are possible between these extremes (e.g., repeated sampling of randomly stacked distributed representations). It seems that the tradeoff between distribution and systematicity is a real one, and synchrony for dynamic binding – although it eases the pain of the tradeoff – is not sufficient to make it disappear completely.

Synchronization and cognitive carpentry: From systematic structuring to simple reasoning

E. Koerner
Honda R & D Co., WAKO Research Center, Chuo Wako-shi, Saitama 351-01, Japan
Electronic mail: *koerner@wrc.honda.co.jp*

I would like to relate this thought-provoking target article by Shastri & Ajjanagadde (S&A) to the dynamic linking by synchronization of rhythmic activity in neural networks. This has been discussed and simulated for sensory segmentation (Shimizu et al. 1985; von der Malsburg & Schneider 1986) and for knowledge-dependent image decomposition and its resynchronization for interpretation in vision, including the development of alternative hypotheses in a quasi time-sharing processing mode at separate phase positions of the rhythmic activity (Koerner et al. 1987; 1990). Whereas those approaches dealt with signal level description and the transition from signal to symbol level, S&A's approach bridges to a more elaborate structuring of the represented knowledge, applying this rhythmic control as a mechanism for easy reasoning in sophisticated knowledge structures, dynamically linking just that part of the stored knowledge that is needed to solve the problem posed. This is the complementary aspect of the above-mentioned approaches (see sect. 2.5).

This contribution bears on the still controversial issue of whether or not oscillatory phenomena in cortical recordings are relevant to an understanding of cortical processing: Yes, oscillations make a lot of sense there. Having dealt with directly related problems from a similar point of view, I agree with S&A in many respects, but instead of simply summarizing all the points I agree with, I will discuss extensions of the conceptual design of the model that are needed to bridge the gap between the signal and the symbol level approach and to give a more detailed description of characteristic aspects of reasoning and decision making in brainlike systems.

I strongly question the statement (sect. 3.4) that there is no need for any central control or system clock. S&A offer no reasonable idea of how such tricky structures can self-organize from unstructured data to allow the emergence of complex knowledge bases at all and to ensure the requisite flexibility, giving the system the chance to modify and create symbols based on persistent subsymbolic descriptions (Smolensky 1988). In this respect learning is not a problem of adjusting weights (sects. 3.4, 10.6) but of self-organizing the algorithmic structure. How is one to resolve conflicts in a limited time in large-scale systems of this fundamentally asynchronous type if there is no helpful demon keeping track of all the locally emerging hypotheses and setting the right phase position? If one has as definite a setup for one's problem as in the proposed model (with presetting of a definite and highly unitary structure, preselected objects, facts, rules, and presetting the proper phase position for each symbolic item to be handled) then the system cannot get stuck but will behave as desired. But how is one to create the appropriate setup to make this approach work so smoothly? My point is that this will turn out to be at least an equally decisive problem in dealing with a "real world problem" like image interpretation.

This is not a problem resulting from simplification that can be resolved in a straightforward way (sects. 9, 10). We have gone the route S&A recommend in section 10.1, and implemented the internal and external scan path as knowledge controlled attention mechanisms to decorrelate the parallel (in-phase) visual input and resynchronize it from asynchronously emerging local hypotheses to an increasingly global consensus, with autonomously ranking alternative hypotheses at different phase positions within the period of global rhythmic control processes (Gross et al. 1992; Koerner & Boheme 1991; Koerner et al. 1987). Synchronization by associative cooperation and local competition as described in section 7.3 will suffice to do the job only for small-scale problems.

At a more realistic scale of both system and problem complexity there is no guarantee of smooth convergence to a consistent global solution (or of any decision at all) in a limited time. Reasoning in such asynchronous(!) systems does not get triggered with well-defined structures in space and time but distributed activation seeds (activated local relational structures) start locally synchronous oscillations (or better, reverberations) that have the aggressive tendency to occupy more systems resources to achieve the activation of the most possible representation. However, with growing system complexity, this is a typical case of combinatorial explosion among alternative decisions.

Exclusively local control is not a solution for this problem, even if we take into account that several alternative decisions can be developed concurrently with the proposed phase labeling. The frequency and phase position of a locally evolving oscillation of a relational structure is defined by its size, structure, and the sensory (or internal) call that triggered it (if you accept the at least partly analog-type evaluation of input activity in neurons and therefore also in neuronal oscillating clusters). Hence, there is not only the range of 40–60 Hz observed in early visual processing (small relational structures), but, with the increasing dimension of the dynamically linked cluster in this aggressive competition for a growing range of dominance, a large variety of irregular frequencies (and of phase positions within these frequencies) emerge. With respect to neural processing we expect this range to be between the highest frequency of about 40–60 Hz (complete matching of inputs to all the requisite eliciting conditions for this parallel represented knowledge structure) and the lowest one (defining the largest possible time interval in which a partly matched representation can self-amplify by synchronizing related representational structures that were not coherently active initially) which we set (for several reasons) to the 4–8 Hz of the hippocampal theta rhythm (Koerner et al. 1990; 1991; submitted). The more simple such a parallel representation is, the higher the probability it will already be activated initially by the complete set of conditions (inputs) and will oscillate with the maximum frequency.

Any such smooth relation between the relative global structure of a representation and its initial frequency of updating is required for stability, so that more global representations with lower updating frequencies will have a chance to take increased control of lower-order representations reverberating at higher frequencies (this is the condition for convergence of the decision process). Hence reasoning with dynamic linking should not be a one-step synchronization; Several time scales are to be expected. The solution is not to replace the definite system clock by a couple of definite frequencies but to allow the emergence of this almost chaotic variety of candidate relational structures in reflexive reasoning (characterized by its frequency and relative phase position) and to guide it to a consisting global solution by monitoring the emerging globalization tendency of coherent activities and by setting a (theta-rhythmlike) adaptive system clock to the most promising phase position. This thereby forces the system to a globally consistent solution by focusing the search on aspects related to this decision.

With reference to experimental evidence and to Minsky's (1985) idea of *A*- and *B*-brain we proposed the hippocampus as this unspecific controller (Koerner et al. 1990; submitted). For such theta rhythm-driven reasoning the limitations on the depth of reasoning do not apply (see sect. 8.2.6).

We too have been attracted by Miller's (1956) 7 ± 2 rule; we accordingly defined the number of alternative solutions the model system should be able to handle concurrently (thereby defining short-term memory). However, we related this measure to the time scale of theta rhythm based on experimental facts more closely connected to the cognitive quality of neural processing than to the observed oscillation in early vision (e.g., the time interval between feedback-controlled saccadic eye movement, behavior-related phenomena in hippocampal theta and EEG recordings, or the statistical distribution of pattern sequence length in human communication, etc.).

Hence, although I agree that this 7 ± 2 story may support such a dynamic reflexive-reasoning scheme, I doubt one can directly and superficially relate observations on an early visual process to the results of a psychological experiment involving much more complex processes and structures.

Reflections on reflexive reasoning

David L. Martin
Computer Science Department, University of California at Los Angeles, Los Angeles, CA 90024
Electronic mail: *martin@cs.ucla.edu*

Shastri & Ajjanagadde (S&A) have taken some important and impressive steps toward understanding aspects of human cognitive capabilities. Although it is still much too soon to know how much of their architecture is genuinely explanatory of human cognition, they have proposed a connectionist system that defines enough architectural and performance characteristics to suggest a broad range of empirical tests and to invite a variety of important extensions. Nevertheless, the system models only an isolated portion of cognition – perhaps artificially isolated – and we need to be clear about what kinds of evidence would serve as meaningful corroboration of it.

Reflexive versus reflective reasoning. S&A's presentation of their system as a model of *reflexive* reasoning only is an approach that is both laudable and troubling. It is laudable because it avoids the common tendency in artificial intelligence work to portray systems of limited capabilities, operating over restricted domains, as if they already capture the essence of some of the least understood, and most general, areas of cognition (McDermott [1981] exposes this tendency well). For example, S&A have wisely refrained from making any claims with regard to the nature of conscious deliberation, which they consider to be characteristic of *reflective* reasoning.

S&A's approach is also troubling, however, because it serves all too well to isolate the model from criticisms of its limitations and deflects many of the questions that are most in need of answers. To any question of the form, "Why can't this system display characteristic X which is clearly present in human reasoning?" it can be answered that X is characteristic of *reflective* rather than *reflexive* reasoning.

Thus, the most pressing need for empirical work related to this model is in corroborating the reflexive/reflective distinction. Given that this distinction holds up, it will be necessary to delineate the boundaries of reflexive reasoning in humans empirically before we can accurately judge the adequacy of this model.

Moreover, the isolation of reflexive reasoning in a model of cognition raises as many hard questions as it deflects. Granted that the distinction between reflexive and reflective reasoning is intuitively appealing, is there sufficient reason to believe that the two depend on mechanisms and representations that are essentially different? If so, how do we account for the apparently smooth integration of the two? How do we account for the instant availability of the products of one for processing by the other, or for the ability to give explicit verbal characterizations of reflexive reasoning, just as we do for reflective? Do the suggested extensions to the model (e.g., function terms and encoding soft and defeasible rules) make sense for reflexive reasoning, reflective reasoning, or both? In learning new rules, what is the relationship between the reflective and reflexive reasoning processes and the representations they use?

What's missing? As S&A have pointed out, there is much that the model does not yet account for, even within the realm of reflexive reasoning. In suggesting areas for further work, it is perhaps most useful to focus on those in which a more complete account would help the most to provide corroboration of the model. Here are a couple of candidates.

Learning. In addition to providing significant new constraints on the form of these representations and a new source of empirical tests, successful learning techniques will be required before the system's ability to scale up to real-world proportions and generality can be demonstrated.

Explaining. As pointed out in section 5 of the target article, facts are retrievable by query processes but not rules or relationships between rules. Thus, it remains to be shown how an explanation of a reasoning process could be given (as in S&A's introductory example of Little Red Riding Hood). (Such an explanation would not necessarily fall within the realm of reflexive reasoning *processes*, but still the *representations* of rules would have to allow for such an explanation.)

Classical versus connectionist architectures. In their influential article, Fodor and Pylyshyn (1988a) questioned the viability of connectionist architectures as models of cognition except insofar as they are used to implement classical models, that is, models that embody compositional representations and structural sensitivity of processes. At the time, unfortunately, the idea of a classical model was roughly identified with completely general mathematical models of symbol manipulation such as Turing machines, and the idea of a connectionist model was roughly identified with relatively unstructured (layered or competitive) masses of neuronlike units. This led to a sense of paradox with regard to connectionist modeling: It was clear that a biologically plausible model had to be connectionist, and yet it was equally clear that a connectionist implementation of something like a Turing machine could not be biologically plausible.

Models such as that proposed by S&A are beginning to suggest the way out of this dilemma by showing that there exists a biologically plausible middle ground based on more structured connectionist architectures that manage to implement limited versions of compositionality and structural sensitivity but fail to approach the full generality of Turing machines. Some of the most interesting psychological work will probably center around the empirical verification of the ways in which humans fall short of this full generality, and some of the most interesting philosophical issues will probably focus on refining our understanding of how such general mathematical models can best contribute to our understanding of intelligence.

What we know and the LTKB

Stanley Munsat
Department of Philosophy, University of North Carolina, Chapel Hill, NC 27599-3125
Electronic mail: *undone@unc.bitnet*

We are forever in debt to the field of artificial intelligence for what we have learned from its failures. Among the legacy of insights from AI is the realization that the understanding of language in all its forms (including stories, jokes, arguments, and explanations of events and actions) requires that we be in a position to bring to bear a virtually limitless array of knowledge and experience – instantaneously. Researchers in AI, being in the business of data manipulation, would quite naturally conceive the problem of "representing our knowledge of the world" and bringing it to bear on the understanding of language as threefold: (1) How do the data get into the system; (2) how are they classified and stored; and (3) how are they gotten to and processed when needed for a particular task (say, reading and answering questions about a particular story). (The three problems are not dealt with independently; each needs to be addressed with the requirements of the other in mind.)

If you suppose that the data base consists of stored propositions ("facts" and "rules"), and that there are tens of millions of them, processing takes awhile. There will be searches through a large amount of data and long chains of processing and so the AI model becomes unrealistic as a model of human language understanding.

Shastri & Ajjanagadde (S&A) have presented us with a connectionist solution to the problem of how large numbers of propositions are stored (the long-term knowledge base, or LTKB) and activated when the time comes (e.g., when a question is posed to the system). The connectionist language understander is much faster because it does not require searches and sequential processing steps in the manner of AI models. But at the same time, the S&A model perpetuates a fundamental assumption of AI models of language understanding. They assume that "what we know about the world" should be thought of as a set of (encoded) propositions. Let me call this assumption *the LTKB-assumption* (or LTKB-A).

I foresee two problems for LTKB-A as a mode for representing what we know about the world and for showing how we bring this knowledge to bear in understanding language. One problem is that the LTKB will contain too much, and the other is that it could not possibly contain enough. It should be noted at the outset that the questions being raised here are not directed at the S&A model per se; their model was never proposed as rich enough to be applied to such problems as story understanding. Rather, the challenges are being offered to call into question the plausibility of *any* model of story-understanding that is based on LTKB-A.

The first problem can be illustrated with S&A's Little Red Riding Hood story. Of all the millions of items in the LTKB that have to do with children, people in general and their behavioral tendencies, people in relation to children, wolves, people in relation to wolves, children in relation to wolves, people in relation to children in relation to wolves, ways of getting hurt, and on and on, how do just the right propositions get invoked (meaning just those propositions that are needed to make sense of the sentence "The wolf heard some woodcutters nearby and so he decided to wait")? One might want to answer, "Well, those are the propositions (actually, one of many possible sets of propositions) that will make sense of the sentence in the story." And that is no doubt true. And *we*, if prodded, can come up with such a set of "sense-making assumptions" out of all the things we know about the world. But short of giving the connectionist network a (homuncular) sense of what it takes to make sense of the story, how does *it* select just the right pieces of background knowledge needed to make sense of the story-sentence? But even this is not the end of the problem. In addition to making sense of story lines, we can recognize when a story line *does not* make sense. Do we explain this as the reader of the story failing to find facts in the LTKB that would make sense of the story? But this will not do either. For story readers can tell you what *would* have to be the case in order for the story to make sense. And they certainly cannot find that in the LTKB. All of this suggests that story understanding is in many respects more akin to story *writing* than it is to fact recalling.

The second problem is that we are capable of bringing so much of our knowledge and experience to bear in language understanding that we cannot have all of that stored as a set of propositions and rules.

Walter and Jane get up for a few dances at a wedding reception. As they return to their table, their friend Jason says, "You two were great. What beautiful dancing partners – you two are just like Fred and . . . " At this point, Rosemary interrupts and completes the sentence: "Ethel."[1] Not everyone will get the joke, but if you do, a reconstruction of the elements of the joke is that the dancing couple is being complimented on their dancing. They are being compared to Fred and someone, so Fred and someone must be a famous dancing pair (famous because their first names alone are enough to identify them). We thus expect the next name to be Ginger. But Rosemary interrupts with "Ethel," thus evoking the pair Fred and Ethel Mertz of the "I Love Lucy" show. Having seen Fred and Ethel Mertz on television, the thought of them as graceful dancers strikes us as hilarious.

Is "Fred and Ethel Mertz would be ridiculous as dancers" in our LTKB? In retrospect, it can seem as if it must be. But how did it get in there? Was it being formulated as we watched the "I Love Lucy Show," as it were, getting us ready to appreciate the joke should it ever be made? If such a proposition is in the LTKB, it is hard to imagine what is not. For example, how about "Fred and Ethel Mertz were not a couple on the Jackie Gleason show"? Is that in the LTKB? If it is, did it get in there as a result of watching the Jackie Gleason show or the "I Love Lucy" show? Again, if such a proposition is in the LTKB, it is hard to imagine what is not. The list of who was not on the Jackie Gleason show is very long. But in conversation someone mistakenly places Fred and Ethel on the Jackie Gleason show (a natural mistake; there was a neighbor-couple on that show, too) and I immediately spot the mistake.

The problem does not just come up in story understanding. Suppose I happen to be in a convenience store, along with several other people, when two men wearing ski masks hold up the store. The next day I am asked questions by a police detective, and I answer them as best I can. How are we to think about where the answers are coming from? One way to think about this is to think of my witnessing of the events of the robbery as producing facts for the LTKB. But what facts? Everything that was true of the robbery that I was in a position to know? The detective asks, "How tall was the one with the green ski mask?" I hesitate. He says, "Was he taller than the cashier?" I immediately reply "yes." Was that fact already encoded in the LTKB?

One reason this seems unreasonable is that if the answer to every possible question about our lives that we can answer is stored as a fact or rule there would just be too many. Second, how are we to suppose that these facts get prized off our experience? What sort of mechanism could take our experience in the convenience store and produce from it *all the facts that there are* in that experience (all the answers to all the questions that I could answer about that experience if asked)? The detective, having spent a career investigating such things, knows just what to ask. Some of his questions may strike me as odd. (Did the one that did the talking come in the door first or second?) Are we to suppose that my inexperienced brain is going to know enough to load my fact bank with the answers to these (and all other possible) specialized questions?

What all of this suggests is that we should not think of what we know about the world as a stored set of facts, even facts distributively coded, which can be accessed as needed. Rather, we should think of experience as more holistically altering the system so that we can *produce* such facts *when the need arises*. But also, alterations in the system (from experience) would have the result that we get on with the reading of a story when past experience supports such facts as *would have to be* the case in order for the story to make sense.

It is, of course, not easy to think concretely about how to model such a system. But then, we are talking about the human mind. Nobody said it would be easy.

NOTE

1. The joke is borrowed from an episode of the television program "Cheers."

Computational and biological constraints in the psychology of reasoning

[a]Mike Oaksford and [b]Mike Malloch
Cognitive Neurocomputation Unit, University of Wales at Bangor, Gwynedd LL57 2DG, Wales, UK
Electronic mail: [a]*pss027@vaxa.bangor.ac.uk;* [b]*mike@cogsci.ed.ac.uk*

Shastri & Ajjanagadde's (S&A's) target article represents a potential milestone in the cognitive science/psychology of human reasoning. Their proposal compels a departure from the more traditional logicist AI perspective, in which the development and implementation of more-or-less formal calculi have been the goals (Braine 1978; Johnson-Laird 1983; Johnson-Laird & Byrne 1991; Rips 1983; but see Oaksford & Chater 1992a). S&A note – from psychological considerations – that humans must continuously compute rapid systematic inferences over very large knowledge bases, but they also note – from computational considerations – that these inferences *must* be of a limited kind and capacity. Human reasoning springs, in their model, not from general-purpose deductive machineries but from the natural dynamics of interacting neural representations. S&A's approach makes an appropriate rejoinder to Fodor and Pylyshyn's (1988b) recent criticisms of connectionism (Chater & Oaksford 1990). They have shown that – within the nonlogicist AI traditions of connectionism, parallel marker-passing architectures (Fahlman 1979; 1981; Hendler 1987), and computational neuroscience (Churchland et al. 1989) – a productive synthesis of psychological, computational, and neurobiological evidence can be brought to bear on the central cognitive problem of reasoning.

In this commentary, we briefly explore the potential of S&A's model to illuminate issues in the psychology of reasoning.

First, by emphasising the need for computationally tractable accounts of human inference, S&A identify an important source of constraint on psychological models (Oaksford & Chater 1992a; 1992b). In cognitive psychology, theories of reasoning have concentrated on empirical adequacy. Constrained laboratory tasks involving at most two or three premises provide the data that these theories attempt to explain (see, e.g., Evans 1982; 1989; Johnson-Laird & Byrne 1991). Ultimately, however, psychological theories must generalise to real human reasoning that may implicate the whole of a person's world knowledge in an inference (Fodor 1983). Current reasoning theories, however, invoke processes that, when generalised to large knowledge-bases, are computationally intractable (Oaksford & Chater 1992a; 1992b). Even if they fully "account" for the empirical data, they *could not* be psychologically real. S&A place the emphasis in just the right place: Realistic theories of human reasoning must not only be tractable, but tractable using biological hardware. [See also Tsotses: "Analyzing Vision at the Complexity Level" *BBS* 13(3) 1990.]

Second, S&A's model appears to generalise naturally to everyday, defeasible inference. The deductive inferences typically investigated by reasoning researchers are computationally intractable by symbolic means, but everyday defeasible inference is worse: The application of a single rule is intractable (McDermott 1986; Oaksford & Chater 1991). Recent claims that at least one extant theory of deduction – mental models (Johnson-Laird & Byrne 1991; see also *BBS* multiple book review of Johnson-Laird & Byrne's *Deduction*, *BBS* 16(2) 1993) – generalises to account for everyday reasoning founders on a problem for which S&A provide a natural solution (Chater & Oaksford 1993; Oaksford 1993). The plausible but defeasible conclusion, from "I turned the key of my car and it has not started," is, "The ignition is faulty." But why is this default conclusion to be preferred to, "The engine has been removed overnight"? The lack of an answer in mental models theory suggests that the real problems of defeasible reasoning are widely unappreciated (Chater & Oaksford 1993, Garnham 1993, Oaksford 1993). S&A show how dynamic bindings and type restrictions can be generalised to provide binding strengths and type preferences that can differentiate between possible defeasible conclusions.

Third, S&A's use of type restrictions may explain one common bias in reasoning tasks (Evans 1989). In "matching bias," subjects tend to ignore negations, instead matching named items (Evans 1972; 1983; 1989). When asked to construct a true instance of the rule, "If there is a blue triangle on the right, then there is not a red square on the left," they may place a blue triangle on the right and a red square on the left (Evans 1972). Oaksford and Stenning (1992) have shown that this bias is due to a difficulty in constructing appropriate contrast-classes. The materials used in these experiments leave the intended contrast-class ambiguous, forcing subjects to match. When the ambiguity is removed, matching bias disappears. Oaksford and Stenning (1992) suggest that type restrictions on a predicate's arguments constrain the contrast-classes identified by a negated constituent. For example, "He did not travel to Manchester by *train*" (italics = rising intonation), identifies *modes of transport* as the appropriate contrast-class because the ternary predicate *travels* has the following associated type restrictions: *travels* (traveller: x; destination: y; mode of transport: z). Typing is of course not unique to S&A's proposal, but we feel that their approach is more likely to generate constrained, tractable typing mechanisms (perhaps explicitly invoking the notion of contrast-class).

Fourth, S&A divide human reasoning into two kinds: *Reflexive* reasoning is rapid, unconscious, and underpins on-line prediction and explanation of the world, anaphor resolution, text elaboration, and so on. *Reflective* reasoning is conscious, and involves external memory aids (pencil and paper) and external representational systems (diagrams, pictures, mathematics, logic, etc.). Other connectionist researchers interested in reasoning (Rumelhart 1989; Rumelhart et al. 1986) have advocated this essentially Vygotskyan distinction that logical reasoning is a function of the internalisation of external representational systems. [See also Hanson & Burr: "What Connectionist Models Learn" *BBS* 13(3) 1990.] This division leaves traditional reasoning theories such as mental logics and mental models without a natural problematic. Those theories seem motivated by requirements and notations from explicit deductive reasoning and are then generalised to reflexive modes of inference (see, e.g., Johnson-Laird 1983). In our view this is misguided: People are reflexive reasoners first; the mechanisms of reflexive reasoning are coopted to perform deductive inference. Unsurprisingly, people are not particularly good at the latter.

Two aspects of our recent work support this position. First, tasks in which deductive performance is poor do not allow subjects to use external aids like pencil and paper. If such reasoning *requires* external aids, then the simple expedient of providing them should improve performance. In some pilot work we supplied subjects with pencil and paper in an abstract version of Wason's (1966) selection task and gave them one minute to solve it. Solution rates were at around 60% compared to only around 4% in the standard task. Second, Oaksford and Chater (in press) have argued that performance on a variety of conditional reasoning tasks can be explained by the reflexive use of a predict-and-explain strategy of the kind that S&A implement in their model.

Finally, S&A's model of reasoning is at the right level to contact neuropsychological investigations of frontal lobe function. Neuropsychological evidence constrains many other areas of cognitive inquiry but not human reasoning. This is anomalous because the impairment of hypothesis testing (Milner 1963) and planning (Shallice 1982) performance (areas also investigated by reasoning researchers) in frontal lobe damage is well known. [See also *BBS* multiple book review of Shallice's *From Neuropsychology to Mental Structure*, *BBS* 14(3) 1991.] We have begun to use standard reasoning tasks with frontal patients (Oaksford et al. 1992b), patients with Parkinson's disease (Malloch et al. 1992), and patients with closed head injuries (Oaksford et al. 1992a), with some interesting results. It may be possible to perform computational "lesion" experiments on S&A's model to see whether qualitatively similar behaviour results. Two factors make S&A's model of particular significance here. First, it is sufficiently well specified to be implemented and to make explicit predictions. This contrasts with the Norman-Shallice (1985) model. Second, S&A's reasoning architecture makes contact with "the rest of" cognition. Computational models of frontal lobe function such as Dehaene and Changeux's (1991), while using biologically motivated building blocks, use architectures that are tied to specific tasks.

The range of implications of theory or model is one guide to its potential for influence. We believe that S&A have provided a technical approach rich in experimental possibilities. This is not to say that it has solved everything; there are several issues S&A have not tackled: They may make too much of functionally questionable neurophysiological results; indeed the "neurobiological" plausibility of their scheme is conceptual rather than factual. Their solution to the variable-binding problem works only for essentially localist representational schemes (or localist views of distributed schemes). Seen as an ingenious use of the time domain to implement marker-passing, their proposal is of course no more (or less!) powerful in itself. Although S&A address the intractability of reasoning over very large data bases, other computational problems well known in AI knowledge representation remain to be resolved in detail (e.g., the frame problem may reappear in the appropriate specification of typing categories). Finally, it is not clear how the rest of a cognitive architecture can "know" (or "learn") how to interact effectively with the particular nodes and oscillations of one of S&A's inference architectures. Despite such objections, S&A's model introduces important new constraints, and a useful expressive vocabulary, to the psychology of reasoning.

This is definitely a step in the right direction on the road to a computationally and biologically, as well as psychologically, constrained theory of human reasoning.

Psychological implications of the synchronicity hypothesis

Stellan Ohlsson
Learning Research and Development Center, University of Pittsburgh, Pittsburgh, PA 15260
Electronic mail: *stellan@vms.cis.pitt.edu*

Unlike researchers who try to prove that symbolic descriptions of human cognition should be replaced by descriptions of neural mechanism, Shastri & Ajjanagadde (S&A) are engaged in the scientifically more fruitful enterprise of relating the two levels of description to each other. The particular computation analyzed in their article is forward inference, for example, to infer the proposition *own(Mary, Book1)* from the implication *give(x, y, z)* → *own(y, z)* and the fact *give(John, Mary, Book1)*. At the symbolic level, such inferences consist of two computations. First, a *match* predicate is applied to verify that the given fact instantiates the antecedent of the implication; the output is the set of variable bindings that make the match true: *Match[give(John, Mary, Book1), give(x, y, z)] = (x/John, y/Mary, z/Book1)*. Second, the bindings are used to substitute constants for variables in the consequent: *Substitute[x/John, y/Mary, z/Book1] [own(y, z)] = own(Mary, Book1)*. The *match* and *substitute* procedures constitute a mechanism for identifying and propagating variable bindings.

The synchronicity hypothesis proposed by A&S claims that the brain performs the match and substitute computations by (1) encoding variable bindings through the synchronous firing of neurons (or clusters of neurons) which represent, respectively, the variable and the constant bound to it, and (2) propagating bindings by linking the neuron representing variable occurrence x' in the antecedent to the neuron representing variable occurrence x'' in the consequent in such a way that if x' fires in a particular phase at time t, then x'' will fire in that same phase at time $t + d(t)$. This mechanism searches through the set of implications in parallel and the time required to infer a particular conclusion is independent of the size of that set.

The synchronicity hypothesis generates two novel psychological ideas. First, S&A explain the limit on working memory capacity as a consequence of the number of temporal phases the brain can keep distinct, the best attempt so far to ground this well-known cognitive limitation in neural mechanisms. The novel idea is embedded in the implication that although there is a limit to the number of *entities* that can be considered simultaneously, there is no limit on the number of *predicates* that can be asserted about those entities. It is not entirely clear how to distinguish between entities and predicates, but this hypothesis might nevertheless bring some clarity to the literature on working memory capacity limitations.

Second, the synchronicity mechanism implies that the brain spreads variable bindings, rather than activation, through long-term memory. S&A add the plausible assumption that the binding information is attenuated with each propagation step; eventually it becomes too fuzzy to support further inferences. Intuitively, this idea differs substantially both from the notion of spreading activation (where activation is a content-free quantity) and from the notion of gradual decay of working memory elements, but formal analyses are needed to verify that these three mechanisms generate different predictions.

S&A also propose a limit on the number of predicate instantiations that can be active simultaneously and a constraint on the syntactic form of inference rules used in backward chaining. These two proposals, however, are not derived from the hypothesized neural mechanism. Both are identified at the symbolic level and motivated with traditional complexity arguments.

The least comprehensible aspect of the synchronicity hypothesis is that it encodes variable bindings in a relation which is *not accessible from inside* the brain itself. That neural cluster *A* is firing in synchrony with neural cluster *B* is detectable by an outside observer, but S&A deny there is any module in the brain that can detect this fact. This takes getting used to. How can a relation which cannot be accessed from inside the system affect further processing? How are the conclusions derived by the proposed mechanism made available to other cognitive processes, for example, planning or decision making?

The proposed mechanism is less integrated into psychological theory than one might have wished. Even as they appeal to behavioral data to support their case, the authors deny that the dynamic storage they are describing can be identified with the working memory studied by psychologists. In order to tell their story, S&A have to introduce what they call an overt short-term memory, an intermediate memory, and an attentional spotlight. No neural mechanisms are supplied for these components and the relations between them and the synchronicity mechanism are left unspecified. Finally, the distinction between reflexive and reflective reasoning, although solidly grounded in behavioral data, comes with some unresolved conceptual questions: Why are there two reasoning mechanisms? Under what circumstances is one or the other mechanism applied? If reflexive reasoning is so efficient, why do people ever resort to reflective reasoning?

In the end, the synchronicity hypothesis may or may not win over alternative hypotheses in the race to account for neuropsychological data. The deeper significance of S&A's article is that it shows that cognitive science is ready to set aside the fruitless debate over whether the mind should be described at the level of neural mechanisms or at the level of symbolic computations and to begin the difficult but important task of specifying how the brain carries out the mind's computations.

Making reasoning more reasonable: Event-coherence and assemblies

Günther Palm
Department of Neural Information Processing, University of Ulm, W-7900 Ulm, Germany
Electronic mail: *palm@neuro.informatik.uni-ulm.de*

Coherence in neural activity can be useful to bind a neural representation together. This is an old idea (Hebb 1949), but still a convincing one. This idea has been elaborated again in recent experimental and theoretical literature concerned with the visual areas of the cortex (experimental papers are cited in the target article; for a discussion of some theoretical issues see Johannesma et al. 1986; von der Malsburg 1986; and Palm 1986). It has turned out that the neural networks in these areas can produce (and make use of?) coherence on a fine temporal scale (msec) in addition to a coarser correlation of activity in the range of tens to hundreds of milliseconds. The former has been called "event-coherence," the latter "rate-coherence." The new idea was that event-coherence could be used to bind the parts (such as edges and corners) of different objects together while rate-coherence would simply indicate that these different objects were presented at the same time in one scene.

Shastri & Ajjanagadde's (S&A's) target article transports this idea from the representation of visual scenes to the representation of knowledge in expert systems: Concept nodes are broadly activated to represent the presence of concepts, whereas event-coherence is used to bind concepts to roles in predicates. In other words, the essential idea is to use fine timing or event-coherence for variable binding.

This is a very appealing idea and it is illustrated quite convincingly in Figures 1 to 13. The idea of combining this system with an *IS-A* hierarchy is also convincing and indeed very useful if not even necessary for any real application of this system.

Only the representation of so-called long-term facts in terms of presynaptic inhibition of inhibitory synapses seems slightly awkward and implausible from a neuroscientist's point of view. One wonders whether the same cannot be achieved by detecting fine coincidence through excitatory synapses. Another problem with the representation of long-term facts is the learning of these facts. During learning these presynaptic inhibitions must be formed somehow, so that axons from cells representing "Mary," for instance, have to find their way to the right terminals connecting the right predicate argument or role ("recipient") to that instantiation of the "give" predicate that was chosen to represent the particular fact that "John gave Book1 to Mary."

Furthermore, one consequence of this fact is that John does not own the book any more, so there should be a way of disconnecting the corresponding inhibitory synapse in the particular "Own" predicate that says that "John owns Book1." The problem is actually even worse, since simple disconnection does not rule out eventually concluding that "John owns Book1" from other facts. So the question remains: How does the inference system deal with negative evidence?

Another problem with the predicate "give" is that "John gave Book1 to Mary," but later Mary may give Book1 back to John. After that, who owns Book1 according to the inference system? Probably both Mary and John. How can the system be prevented from drawing both conclusions? I believe it would be much more reasonable to store as long-term facts not the propositions about "giving" but rather the resulting propositions about ownership. Thus some of the inferences should actually be drawn before storage and then stored as long-term facts. Incidentally, the use of the term *long-term fact* is also a bit misleading, because it apparently need not mean a fact in long-term memory but rather a fact that is memorized during the apprehension of a short story. The inference system presented here is clearly oriented toward answering queries about short stories rather than constituting a consistent complete world-model as in long-term memory. Thus the target article does not address the problem of what to store and what to infer, which is fundamental for the organization of large data bases as well as for the understanding of human long-term memory.

Despite these obvious problems, most of which are not specific for the idea presented in the target article, S&A illustrate quite convincingly how synchrony can be used for dynamic binding. Toward the end of their paper, however, when it comes down to the technical problems (sects. 5 & 6), the nice idea gets marred with a number of strange and clumsy constructions, in particular the "up/down switches" and the "predicate banks." I wonder whether one could perhaps get along without these constructions.

Predicate banks. I think the idea of an *IS-A* hierarchy should be extended to the predicates: There is a general "give" and under it there is a special "give" representing "John gave Book1 to Mary." Systematic relationships as in Figure 6 should be represented between general predicates ("give," "buy," "own," etc.) and not between their special instantiations as in Figure 12. The reason for this is simply that the connections between "give," "own," and so on should be implemented only once (between general predicates) and not between all the different instances of "give," "own," "buy," that may be represented as long-term facts. Thus Figure 12 should contain a general "own" ellipse between "can sell" and "own" and a general "give" ellipse between "own" and "give." Furthermore, the connecting paths should be from "can sell" (which is general) via general "own" to particular "own," and from general "own" via general "give" to particular "give." This would not change the arguments given in S&A's paper, it would only increase the length of the shortest path by two steps. Using this kind of *IS-A* hierarchy also for the predicates would be a simple alternative to S&A's introduction of "predicate banks."

Up/down switches. I think these switches can also be replaced by a more plausible and perhaps simpler mechanism if one uses a distributed representation of the concepts in terms of Hebbian cell assemblies (Hebb 1949; Palm 1982; 1990) instead of single nodes. In this framework it is conceivable to represent an *IS-A* hierarchy (of concepts or predicates) in terms of set-containment of the corresponding assemblies (sets of nerve cells).

For definiteness let us assume concepts that are higher in the hierarchy are represented as smaller assemblies. Then upward inference could be performed by raising the average threshold of neurons in the network (Palm 1982), thus forcing the representation to become sparser. Conversely, downward inference could be performed by lowering the threshold. Furthermore, the use of cell assemblies for the representation of concepts makes it possible to represent similarity between concepts in the degree of overlap between the corresponding assemblies.

Another improvement of the proposed inference system could be the use of more than only binary logical values for the certainty of propositions. One could represent the certainty or confidence for a proposition by means of the rate of firing of the corresponding unit. This is a little problematic with the model proposed in the target article, because it uses phase-coherence with respect to a fixed frequency to represent binding. The more general idea to use event-coherence (vs. rate coherence), as mentioned in the beginning, does not have this problem.

Thus a number of technical problems can perhaps be solved and the representational scheme improved considerably by using cell assemblies instead of single units for the representation of concepts and event-coherence instead of phase-coherence for the representation of bindings. These ideas would of course have to be worked out more accurately, but I believe they could help to make the proposed system more amenable to neurobiological theorizing and perhaps even more useful in practice.

I also found the target article very useful for triggering thoughts on the practical use for event-coherence – perhaps even in the visual cortex.

Useful ideas for exploiting time to engineer representations

Richard Rohwer
Department of Computer Science & Applied Mathematics, Aston University, Birmingham B4 7ET, England
Electronic mail: *rohwerrj@cs.aston.ac.uk*

Shastri & Ajjanagadde (S&A) have found an interesting way to exploit the representational potential of time in neural network models. In most "neural software engineering," a correspondence is defined between some of the state vectors of the model and interpretations in an application domain. The representational power of a state is limited by its dimension; for example, a network of N binary-valued nodes can represent at most 2^N different things. But without allocating any further hardware resources, that representational power can be increased to 2^{NT} by interpreting length-T temporal sequences of states instead of individual states. It is a space-time trade-off: It takes T times longer to represent something this way, but 2^T times as many things are representable.

S&A have found a situation in which this trade-off is an impressively good deal. It is important to have the power to represent a great variety of variable bindings but most will never actually get represented in practice, and most of those that do will not need to be represented for very long. Hence, it is better to spend some time rebuilding the representational setting each time a binding needs to be represented than to keep lots of spare representational capacity on tap.

The space-time trade-off in this system is partly illusory, because its dynamics is order $T - 1$ in the state variables, where T is the number of phases in a fundamental period. This is because maintenance of synchrony requires connections with time-delay $T - 1$ between the ρ-btu nodes representing corresponding parts of rule-related predicates. Consequently, so far as the dynamics is concerned, a "state" has $N(T - 1)$ components. Whether temporal synchrony is implemented with simple delay lines or the elaborate mechanism in S&A's section 7.3, a buffer of size $N(T - 1)$ has to be directly or indirectly implemented for the system to run. These extra degrees of freedom can be thought of as implemented at a subcellular level. Computer simulations have to dedicate memory to them.

Although temporal coincidence plays a key role in this system, the oscillations seem inessential to its operation. What matters is that fact predicates "observe" whether their arguments fire synchronously with any constants at least once during a reasoning episode, and that variables linked by rules eventually fire at the same time as any constant to which they may be bound. Periodic reiteration of these coincidences seems a waste of time. The only important role of the oscillations is in keeping variables linked by rules synchronised with each other. That way a constant synchronised with one is synchronised with all. The synchronisation among rule-related variables would be maintained by instantaneous propagation of activations, if only that were possible. Instead, it is achieved (eventually) by delaying propagation for nearly one basic oscillation period, or by more elaborate mechanisms that require at least one cycle to take effect. Perhaps there is a cheaper way.

This system's elegant distribution of representations over time is not matched by an elegant distribution of representations over nodes. Grandmother-cell (or cell cluster) representations of constants and variables are used throughout. This may be just as well for expository purposes, but greater efficiency and potentially interesting properties may arise from more fully distributed representations. A set of C constants, for example, can be represented as patterns distributed over $O(\log C)$ nodes. (A sparser representation using $\alpha \log C$ nodes, with $\alpha \gg 1$ but nevertheless $\alpha \log C \ll C$, might have more useful properties.) Smolensky, Dolan, and others have developed "tensor product" binding methods that use distributed representations of constants and variables (Dolan & Smolensky 1989). Unfortunately, these methods require $(\alpha \log C)(\alpha \log V)$ nodes to represent bindings among C constants and V variables. C and V refer to *all* constants and variables, not just those used in an episode of reasoning. It seems feasible, however, to distribute the tensor product over time, using a mixture of the tensor product binding and phase binding approaches (Rohwer 1993). This offers the combined advantages of each system. The total number of nodes required to represent the constants and variables is reduced from the grandmother-cell system's $O(C + V)$ to $O(\log C + \log V)$. No extra nodes are needed to represent the tensor product, but some extra time steps are needed, as many as there are bindings in the episode of reasoning. In addition to providing increased efficiency, the distributed representations might give such a system interesting generalisation properties found in the more popular neural network models.

Do simple associations lead to systematic reasoning?

Steven Sloman
Department of Cognitive & Linguistic Sciences, Brown University, Providence, RI 02912
Electronic mail: *sloman@cog.brown.edu*

This is an interesting model that is consistent with several common intuitions about human reasoning. The image of parallel chains of inference unfolding and refining themselves over time with related elements bound together through phase synchrony is appealing. Despite a widely shared belief that many of our mental representations have an intrinsically sequential character (e.g., our memory for music), few models have succeeded so well in using time as a representational device. Unfortunately, the model is devoid of empirical support. It is so rich in assumptions and detail that vast quantities of confirming data would be required for it to merit serious consideration. And the little that is known about human reasoning makes such data unlikely.

The chief source of evidence appealed to by Shastri & Ajjanagadde (S&A) is neurophysiological. They argue vehemently for the model's "neural plausibility." But the data they depend on for this vague claim are disconnected from the domain they are modeling. The strongest evidence they muster is a suggestion that "the dynamic binding of visual features pertaining to a single object may be realized by the synchronous activity of cells encoding these features" in the cat visual cortex. (sect. 7.1.1). Even if we accept this evidence at face value, does it tell us anything at all about how people reason? The binding of object features may depend on temporal synchrony, but the model posits a particular parametrized temporal synchronization process that binds the arguments of abstract predicates to their fillers and instantiates the processing of abstract chains of inference. The data are so far removed from the domain of study that even S&A admit that the only relation is analogical. Whether or not the brain makes use of temporal synchrony in object perception has no bearing on how we reason abstractly, especially because we can guess only roughly at what either of the underlying psychological processes are. S&A lift a rich and promising metaphor (binding through temporal synchrony) to the status of scientific evidence. This could be more easily ignored if it were not so basic to their argument.

S&A also report evidence suggesting that their model allows them to predict working memory capacity. But many more firmly grounded theories already account for working memory data (e.g., Baddeley 1986).

A model of human reasoning should presumably have, first and foremost, implications for human reasoning. Accordingly, S&A make a few relevant predictions (sect. 8.2.6) but supporting data or even suggestive examples are not provided. Some of the model's predictions seem rather arbitrary and therefore unlikely to be confirmed, especially those having to do with restrictions on when variables can appear in the antecedent or consequent of a rule. Other predictions are just misguided. For example, the model predicts that people can make transitive inferences using only a small number of relations, but examples of transitive inference that pose no difficulty for people can be constructed easily, even from esoteric relations. Consider the transitive sequence *Pimplier(teenager$_1$, teenager$_2$)* & *Pimplier(teenager$_2$, teenager$_3$)* & . . . & *Pimplier(teenager$_{n-1}$, teenager$_n$)*. The inference *Pimplier(teenager$_1$, teenager$_n$)* can be drawn effortlessly (even reflexively). People are terrific at constructing linear orderings when the context clearly calls for one.

The model's problems begin with its failure to capture aspects of people's fallibility that much simpler connectionist models are able to capture easily. For example, people do not always respect the logical principle of category inclusion. To illustrate, when evaluating the strength of an argument of the form "*premise statement*, therefore *conclusion statement*," people often fail to take inclusion relations between premise and conclusion objects into account. For example, they fail to judge an argument such as "*Animals* use norepinephrine as a neurotransmitter, therefore *reptiles* do" as perfectly strong. In fact, on average they judge it substantially weaker than the argument "*Animals* use norepinephrine as a neurotransmitter, therefore *mammals* do." A connectionist network much simpler than the one described in the target article provides a straightforward account of this finding (Sloman 1993). People seem (in their reflexive state) to reason less in accordance with many of the rules of logic or *IS-A* hierarchies than they do with heuristics that depend on similarity, metaphor, and the surface structure of statements (see, e.g., Klayman & Ha 1987; Lakoff 1987; Wason 1960).

Examples of people's tendency to rely on similarity over logic are found in demonstrations of violations of the conjunction rule of probability. The conjunction rule states that because the extension of the conjunction of events *A&B* necessarily includes the event *B*, $P(B) \geq P(A\&B)$. This rule is violated in that, for example, people who are given a description of a man who is intelligent, unimaginative, compulsive, and generally lifeless are more likely to infer that he is an accountant who plays jazz for a hobby (*A&B*) than they are to infer that he simply plays jazz for a hobby (*B*), apparently because the description is more representative of *A&B* than it is of *B* alone (Tversky & Kahneman 1983). The point is not that an account of these particular phenomena could not be generated using the representational scheme of the model described in the target article, but rather that the model gives us no a priori reason to expect these basic characteristics of human reasoning. The model serves as an existence proof that a network of nodes and links can use temporal synchrony to traverse an inferential dependency graph. But many interesting qualities of human reasoning are not explained by such a graph.

To argue that such systematic errors are the result of the intrusion of *reflective* processes on a *reflexive* process that is otherwise logical is just the opposite of what we should expect from psycho-logic. Let us hope we can put more faith in conclusions we come to upon reflection, on the assumption that our quicker, dirtier reflexive thinking will sometimes be wrong.

One common approach taken to at least motivate the empirical validity of an elaborate model that rests on many assumptions is to show that it is able to account for some interesting set of data. The model would be much more convincing if S&A showed that it could in some sense comprehend the Little Red Riding Hood story with which they begin their discussion, or even some simpler story. Without even this kind of empirical support to buttress the model, the target article leaves us with little other than a potentially promising metaphor to describe human reasoning. But because the mind is so good at generating metaphors, a new metaphor is not something the study of the mind really needs.

Phase logic is biologically relevant logic

Gary W. Strong
College of Information Studies, Drexel University, Philadelphia, PA 19104
Electronic mail: *strong@duvm.ocs.drexel.edu*

The target article by Shastri & Ajjanagadde (S&A) presents an exciting new model of binding-dependent logic (phase logic) that is consistent with some very basic human information-processing limitations. Their interpretation of Miller's *magic* number 7±2 (sect. 8.2.3) in terms of binding limitations of a synchronous oscillatory system provides a ground-breaking link between this well-known limitation of human cognition and underlying neural architecture. In addition, S&A have clarified, if their model holds up, the well-known paradox of why it is so difficult for novices to become experts upon being presented with rules derived from expert behavior. Until rules become part of the long-term knowledge base (LTKB), they must be processed reflectively rather than reflexively, and, in order for them to become part of the LTKB, the rules must have been relevant in a number of cases (sect. 8.5). Overall, S&A's phase logic model offers an intriguing alternative to traditional AI approaches such as symbol rewrite systems, showing how biologically relevant models can exhibit a systematic correspondence between arguments of first-order predicates and the appropriateness of argument correspondence in terms of class membership. It is a shame that S&A did not report the simulation results they mention in section 10, because they could have been more convincing in their arguments, having shown how they dealt with the details of instantiating their model while preserving biological realism.

A small criticism I have of the target article concerns the way S&A interpret the logic circuit of their fact encoder. Their circuit for encoding *give(John, Susan, x)* will not recognize *give(John, Susan, Car7)*. Their interpretation of *give(John, Susan, x)* is "John gave Susan something," which is inconsistent with the closed world assumption (CWA) the authors assume in section 4.4. The CWA requires that a "don't know" answer be viewed as a no answer and it implies a failure to recognize *give(John, Susan, Car7)*. A proper circuit for encoding the fact "John gave Susan something" is not the one S&A illustrate but one that makes use of their *IS-A* hierarchy by connecting an abstract object to the τ-and node as an inhibitor of the *g-obj* line. A description of such an implementation would clarify their interpretation of unbound arguments in phase logic.

I have a more substantial criticism of S&A's claim that nodes cannot bind with more than one entity at the same time. For example, in section 4.8 they claim that the node *Animal* cannot fire in synchrony with both *Tweety* and *Sylvester* at the same time. In a periodic phase logical system this may be a reasonable claim but there is no reason to require that phases be periodic. This unnecessary claim led S&A (in sect. 5.2) to what I believe is an implausible architectural feature, that of concept clusters, each with k_1 banks of ρ-btu nodes, where k_1 is the *multiple instantiation constant* and refers to the number of dynamic instantiations a concept can accommodate. The Strong and Whitehead (1989) simulation, to which S&A refer, showed that, with an appropriate architecture of spiking neurons such multiple bindings *are* possible. Our simulation demonstrated cyclic activity in overlapping subsets of *minicolumns* (Strong & Whitehead 1989, p. 396). The architecture we used includes, as basic processing units, minicolumns that contain an ensemble of neurons that share inputs. Such sharing also allows the ensemble to achieve independence from absolute refractory periods of individual neurons as well as from "noisy propagation delays." The latter is an implementation problem recognized by S&A (sect. 7.3) and handled through group averaging within ensembles.

The use of mutually inhibitory minicolumns as basic processing units caused the Strong and Whitehead (1989) simulation to exhibit a behavior that obviates another biologically implausible requirement that S&A claim is necessary for their model: Entities used in argument bindings must have specific delays associated with them (sect. 4.3) to produce a phase separation between different entities. Whereas a central control (such as the hippocampus) might play a role in separating entities (as suggested by Eichenbaum et al. 1989), S&A have ruled this out. Strong and Whitehead's simulation demonstrated that entity separation can be achieved, in any case, without central control and without concept clusters, *even with* overlapping node sets. This can be seen in our simulation output, where the boundaries between phases are fairly sharply defined (see Figure 1, which shows a more recent sample of our simulation output).

In sum, S&A's model is a very important contribution to the field of cognitive science in helping to bridge the logic-based approaches of traditional AI and biologically relevant models of human cognition. They have constructed this bridge with a model whose architecture explains some very basic limitations on human information processing. With a bit more attention to the details of actual neural processing units, however, the power of their model may be substantially improved. I suspect that they encountered this need in the simulations they indicate they conducted.

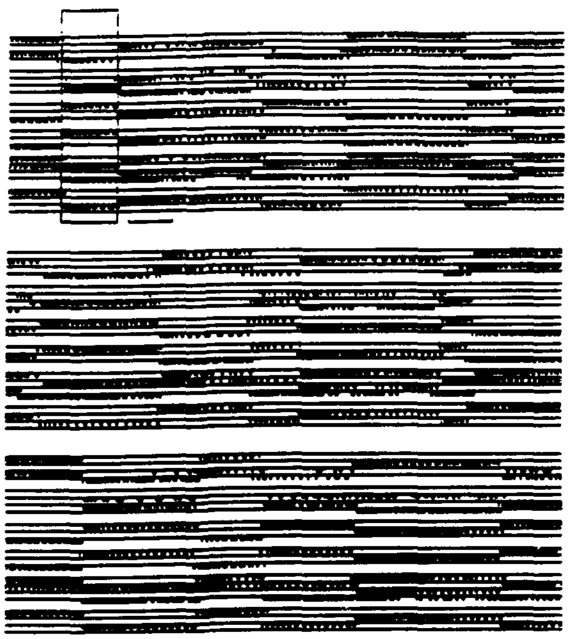


Figure 1 (Strong). "Three staffs" of simulation output during a free-running mode of operation following the learning of three overlapping patterns. Each line represents the output of one minicolumn. There are three different "phases" of output activity that bind different subsets of minicolumns, and the transitions between the phases are fairly well defined. The box surrounds the second phase, which has one minicolumn in common with the first phase. The phase that follows the box has one minicolumn in common with the boxed phase.

Temporal synchrony and the speed of visual processing

Simon J. Thorpe
Institut des Neurosciences, Département Neurosciences de la Vision, Université Pierre & Marie Curie, 75005 Paris, France
Electronic mail: *thorpe@ccr.jussieu.fr*

Shastri & Ajjanagadde (S&A) have provided a remarkable example of how ideas in cognitive science can converge. When I first heard Lokendra Shastri describe his work with Venkat Ajjanagadde on using temporal synchrony to tackle the dynamic-binding problem in 1989, they were approaching the problem as computer scientists. At about the same time, the work on oscillatory activity in the visual cortex was beginning to cause quite a stir in the neuroscience community (Eckhorn et al. 1988; Gray et al. 1989), but at that point these two different approaches appeared quite separate. In the spring of 1990, however, a meeting on temporal coding in Paris provided an opportunity for interdisciplinary discussion and the cross-fertilization of ideas. Clearly, S&A have taken the interdisciplinary challenge seriously and have worked hard to make their model consistent with neurophysiological and psychological data.

I have some more specific points. S&A devote nearly all of their target article to an analysis of language understanding with an appeal for experimental data to test their ideas (sect. 8.2). Although psychological data on language comprehension are available, it will probably be quite difficult to test the neurophysiological plausibility of such a model in this domain, because so little is known about the neuronal activity during language understanding (though see, e.g., Creutzfeldt et al. 1989). S&A do mention (sect. 2.5), however, that similar problems of dynamic binding arise in vision. Indeed, the connectionist model recently developed by Hummel and Biederman (1992) also uses an approach based on synchrony of activation to tackle the problem of binding elements during shape recognition. Can detailed knowledge about visual system function be used to test the feasibility of S&A's model of language understanding?

A few years ago we pointed out the serious computational problems posed by the remarkable rapidity with which the visual system can process images (Thorpe & Imbert 1989). We argued that processing was so rapid that a great deal of processing must be possible on the basis of only one or at most two spikes per neuron. The argument was as follows. There have been a number of reports of neurons in the monkey temporal lobe with responses selective for complex visual stimuli such as faces. One of the most remarkable features of such neurons is that they typically respond only 100 to 140 msec after stimulus presentation. On the basis of anatomical studies it would appear that such neurons are at least 10 synapses away from the photoreceptors of the retina (information has to go through LGN, V1, V2, and V4 en route), which implies that each processing stage has only approximately 10 msec before the information has to be forwarded to the next layer. Since the firing rates of cortical neurons rarely exceed 100 to 200 spikes per second, this means that even if visual processing involves essentially feed-forward processing, much must be achieved on the basis of only 1 or 2 spikes per neuron.

The strength of this argument has been considerably enhanced by some recent data of Oram and Perrett (1992), who looked at the time course of the face selectivity of neurons in the primate temporal lobe and reported that selectivity is fully present during the first 5 msec of the neuronal response even for neurons with onset latencies of less than 100 msec. Other data on neurons at earlier stages in the visual system from our own laboratory also found that selectivity was typically present right from the very start of the neuronal response, both in the case of orientation selectivity (Celebrini et al. 1993; Thorpe et al. 1989) and selectivity to stereoscopic disparity (Thorpe et al. 1991). Such data imply that information processing must be very rapid.

Oram and Perrett conclude that "the only way to account for the rapid discrimination is to consider a coding system in which the first spike from multiple sources is used to transmit information between stages of processing" (p. 70).

But if processing of even complex stimuli such as faces can be achieved on the basis of only one or perhaps two spikes per neuron, what of S&A's model, in which several cycles of synchronized activity are required to allow reasoning? One response would be that although there is clear evidence for a hierarchically organized architecture in the case of the visual system – cells in the temporal lobe are something like 10 synapses away from the retina – the same is probably not true for the neural structures involved in language understanding. Thus while synchronous firing may be difficult to obtain across the different hierarchical levels involved in visual processing (at least for the rapid visual processing that leads to the activation of face-selective neurons in the temporal lobe), this may not be a problem for language processing.

It may be that the visual system does not actually need oscillatory activity to establish the sort of grouping-related synchrony of firing required by S&A. One of the surprising results of recent studies of visual response latencies in the visual cortex is the remarkable range of onset latencies found. Under the same stimulation conditions, some visual cortical neurons will start firing 40 msec after stimulus onset whereas others have onset latencies of over 100 msec (Celebrini et al. 1993; Thorpe et al. 1989; Vogels & Orban 1991), with most cells starting to fire with latencies between 50 and 70 msec. This range of latencies is sufficiently large to mean that synchrony can be used to group subsets of neurons even in the absence of oscillatory activity. For example, one set of features could be grouped if the relevant neurons fired around 50 msec, whereas another set of features could be grouped by having the relevant neurons fire around 60 msec. As Eckhorn (1991) and others have already pointed out, stimulus-induced synchrony may provide an alternative way of grouping features without the need for oscillatory activity.

In conclusion, the use of stimulus onset induced synchrony in the visual system may allow the operation of a feature-binding process similar to the one proposed by Shastri & Ajjanagadde. The major difference is that it can potentially work even in feedforward networks and does not require the use of oscillatory activity, a significant advantage given the difficulty that a number of researchers have had in demonstrating oscillatory responses with static visual stimuli (see, e.g., Tovee & Rolls 1992).

Should first-order logic be neurally plausible?

David S. Touretzky[a] and Scott E. Fahlman[b]

School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213-3891

Electronic mail: [a]*dst@cs.cmu.edu;* [b]*sef@cs.cmu.edu*

Part of the attraction of NETL (Fahlman et al. 1983), but also a source of difficulty, was that it conflated a theory of representation with a parallel computing architecture. The emphasis on parallel implementation discouraged people from looking too closely at the representation ideas. Shastri & Ajjanagadde (S&A) appear intent on repeating this mistake, with claims of neural plausibility further clouding the issue. Their representation is not very humanlike – but notice how well it fits the computing architecture. Their architecture is not very brainlike – but look at the complex inferences it supports. Unfortunately, when teased apart and critically examined, neither component holds up as a credible proposal about human cognition.

Let us first consider the representational component. Humans can make certain kinds of inferences very rapidly, an observation that has motivated many proposals over the years, including NETL. S&A want to identify human reflexive reasoning with a restricted subset of Horn clause logic. But where is their evidence in support of this claim? The only shred of justification we can find is that they have, lurking in the wings, an elaborate scheme for wiring up an implementation.

Human reasoning does not seem to have much in common with the type of inference at the core of the S&A proposal. S&A tackle this difficulty in two ways. In sections 1.4 and 9.1 they exclude entire classes of phenomena – analogical reasoning, episodic memory, imagery, and associative recall by fast set intersection – that would appear to be more central to human reasoning than strict logical deduction (Lakoff 1987; Lakoff & Johnson 1980). But in section 5.5 they promise something a little more flexible than *modus ponens*, namely, soft rules – vaguely defined and unimplemented – and defeasible inference. We also note a stab at abduction in Ajjanagadde (1991).

These crude initial forays into an area far more complex than logical deduction are no substitute for a credible theory of human informal reasoning, namely, something comparable in scope to Collins and Michalski (1989). Thus, the crux of the S&A theory remains a certain restricted, first-order logical language put forth as a language of the brain, with nothing to recommend it – as a representational theory – over far more sophisticated nonconnectionist proposals.

The second component of S&A's proposal is a supposedly neurally plausible implementation, a claim that falls apart almost immediately. It is fine to propose synchronous firing as a binding mechanism, but this feature cannot serve as the sole biological justification for what turns out to be a complicated parallel computer architecture.

As a neural model, the S&A proposal suffers from multiple fatal flaws. It is essentially localist, postulating disjoint neural populations for distinct concepts. Yet it is also highly redundant, requiring multiple copies of any concept that might participate in more than one simultaneous relationship. These multiple copies are controlled by a switching mechanism whose wiring, in terms of complexity and specificity of connections, is unlike any neural circuitry described in the literature. We also question whether a system that requires oscillation with up to ten separable, stable phases is any more neurally plausible than naive "computer in the head" models. Certainly, nothing like this precise time-keeping has ever been observed in brains.

We conclude with some remarks on the relation of this work to the NETL system. We see nothing in S&A's proposal that cannot be done by NETL, which also exhibited some additional abilities such as *set-intersection*. It is a straightforward operation in NETL to include statements like *scared-of(y,x)* in the prototype definition for the *preys-on(x,y)* relation, and to inherit the former whenever the latter is asserted for specific individuals. The S&A model can be extended to pass fuzzy quantities rather than discrete markers, but so could NETL (Fahlman et al. 1983).

NETL had a central controller that told the knowledge representation network what to do on a cycle-by-cycle basis. S&A observe that this is not neurally plausible and claim that a significant contribution of their work is to show how such a system can run without a central controller. This claim is stated repeatedly, but it reminds us of the Wizard of Oz: "Pay no attention to the man behind the curtain." We believe that the authors have hidden the controller, not eliminated it.

It is true that the S&A proposal does not require cycle-by-cycle control. Instead of having a controller step markers upward or downward through the type hierarchy, S&A create two distinct parallel networks. One propagates phase-based markers upward from a starting point and the other passes markers downward. Some external agency must still select which of these networks is to be active, but once this has been done, the marker propagates as far as possible in the specified direction. The authors can fairly claim that they have eliminated cycle-by-cycle control but at the cost of further replication of network hardware and the loss of certain useful operations that require more precise control.

In any case, there remains a need for some agency to manipulate the many control lines, set certain nodes oscillating with the same phase, query the network via the appropriate *e* and *c* lines, and so on. In section 10, S&A suggest that this is not done by a "controller" but rather by the "parser." It appears that the central controller has been eliminated by distributing a few minor elements and renaming all the rest.

Dynamic-binding theory is not plausible without chaotic oscillation

Ichiro Tsuda
Department of Artificial Intelligence, Kyushu Institute of Technology, Iizuka, Fukuoka 820, Japan
Electronic mail: *tsuda@dumbo.ai.kyutech.ac.jp*

Dynamic binding of knowledge is one of the essential processes for both "reflexive reasoning" and "reflective reasoning" in the human cognitive system. Shastri & Ajjanagadde (S&A) deal with reflexive reasoning in terms of connectionist models of dynamic binding. This approach may assure a plausible model of the process of dynamic computations. Indeed, S&A propose a reasonable model of reflexive reasoning. To justify the model biologically and from the viewpoint of dynamical theory, however, they refer to the synchronization or phase locking of periodic oscillations that was observed in the visual cortex of the cat (Eckhorn et al. 1988; Gray et al. 1989) and monkey (Kreiter & Singer 1992). This commentary is devoted mainly to the question: Could synchronized or phase-locked periodic oscillations provide a plausible basis for the dynamical model of reflexive reasoning?

In the Gray et al. (1989) experiments, rapid damping of both auto- and cross-correlation functions was found. There are two possible causes of the damping: One is due to inherent chaos, and the other is perturbation by noise. The latter possibility can be rejected. We see an apparent feature of the observed correlations, namely, time symmetry of autocorrelations but time asymmetry of cross-correlations (see Fig. 1–3 in Gray et al. 1989). When the periodic oscillation is perturbed by noise, cross-correlation between two such oscillators should be symmetric in time as well as in autocorrelations because of the statistically stationary motion. The assumption of the existence of chaotic oscillators, however, leads us to a reasonable explanation of the distinct feature of correlations that a transient process accompanied by desynchronization between chaotic oscillators brings about time asymmetry of cross-correlations, preserving time symmetry of autocorrelations due to inherent stationary chaotic motion.

In addition, we examine whether or not desynchronization can be achieved by noise, since there is still a possibility of the participation of noise in the transient process, which may give rise to asymmetric cross-correlations. Desynchronization is due to a separation of corresponding orbits and the degree of separation can be measured by the degree of orbital instability indicating an exponential separation of nearby orbits. The Lyapunov exponent is the average rate of this separation in unit time. Since desynchronization should start unless all components of the Lyapunov spectrum are negative, the value of the nonnegative Lyapunov exponents determines the degree of desynchronization. The contribution especially of the largest one, λ, will be dominant. It is reasonable that the time necessary for desynchronization is of the same order as the inverse of the largest Lyapunov exponent, $O(\lambda^{-1})$, since λ^{-1} is the time necessary for the *e* magnification of a tiny initial separation. If noise participates in desynchronization, infinite time is theoretically needed for desynchronization, since in the case of noise λ is zero.

Thus, we conclude that the cause of the damping is the existence of inherent chaotic oscillators. At the moment it is difficult to estimate the correct value of the orbital separation of the neural oscillations; it seems plausible, however, to estimate it as the order of one per one cycle of oscillation. Hence, as an order-estimation, the desynchronization takes 20~25 msec. Taking into account a cut-off-frequency of around 100 Hz in the experiments, the unit time of the observed oscillations should be 10 msec. Then λ is estimated at around 0.5 per unit time, which is a reasonable value from the viewpoint of dynamical theory. Thus, the reasoning of S&A must be amended in its "biological interpretation" of their theory. Actually, our preliminary numerical simulation of the chaotic model for cortical neuro-oscillations shows much faster desynchronization than the theoretical estimation. In most cases, a time less than one-half cycle of oscillation is required.

The neural (de)synchronization is a more rapid process, so the synchronized state cannot be sustained for the few hundred milliseconds supposed by S&A. It is plausible that the neural synchronization makes rapid judgments by feature detection (Gray et al. 1989), or by initiating cognitive processes (Koerner et al. 1987). Throughout the process of thinking, including "reflexive reasoning," a chaotically itinerant motion among "attractors" (we call it "chaotic itinerancy") seems much more plausible, one that can generally appear in systems with large degrees of freedom (Davis 1990; Ikeda et al. 1989; Kaneko 1990; Tsuda 1991). In such itinerant motions, the system is temporally expressed as a "small" system, where "small" means the participation of only a few dominant modes accompanied with a number of inactive modes that could be active at the next period of the process. These modes can be activated as a chaotic mode by a large number of interactive neurons (Freeman 1987; Skarda & Freeman 1987). The temporal reduction of the number of active modes must stem from spatial coherency (Freeman 1991), but not from "phase locking."

Related to the above discussions, I would also like to comment on the possibility that von der Malsburg's model for the cocktail party effect (von der Malsburg & Schneider 1986) has nothing to do with the neural synchronization observed in the experiments. The cocktail party effect is more dynamic and complex, hence its explanation needs such a mechanism of dynamic information processing that both coherence in space and chaotic itinerancy in time play a role in sustaining memories during a period of a few hundred milliseconds to a few seconds, and in searching and linking appropriate items in the LTKB (long-term knowledge base). Here, spatial coherence is necessary for the dynamic link of neural activities over wide cortical regions, especially related to auditory processing, short-term memory, and thinking. Chaotic itinerancy creates the dynamic sustaining of memories and the processing of meaning, namely, a dynamic link of memory items (Tsuda 1991). We have shown that a coupled chaotic system and a chaotic neural network, which can exhibit chaotic itinerancy, sustain any information fed from outside by means of propagating local chaotic activities despite the elementary chaotic process (Matsumoto & Tsuda 1987; Tsuda 1992).

In addition, I recommend S&A to the following literature concerning the roles of neural synchronization. It has been hypothesized, for example, that synchronization of neural oscillations may participate in the processes of rapid interpretation, image synthesis, relation formation in knowledge base, and parallel byte-formation in the sequential flow of visual information (Holden & Kryukov 1991, Koerner et al. 1987, Reitboeck et al. 1990; Shimizu & Yamaguchi 1987). Furthermore, concerning dynamic features in coupled oscillator systems, studies by "coupled map lattices" (Kaneko 1989) and "phase dynamics" (Kuramoto 1991) should not be overlooked. The latter concerns mainly a periodic and synchronized regime, and the former treats various kinds of complex dynamic regimes.

ACKNOWLEDGMENT
I thank Walter Freeman for stimulating discussion.

Ethereal oscillations

Malcolm P. Young
University Laboratory of Physiology, Parks Road, Oxford OX1 3PT, England
Electronic mail: *mpy@physiology.oxford.ac.uk*

When I was a first-year undergraduate psychologist I developed an interest in the idea that cognition might have something to do with the brain, and was promptly dispatched to the department of physiology to learn something about my chosen organ. After hearing a bit about nerve cells, impulses, and such, I produced a gigantic first essay in which I tied up what I had learned of physiology into an account of how the brain worked in what, I was convinced, was a monumental achievement in quality as it was in quantity. As it happened, my unfortunate tutor, an eminent Spanish nociception researcher, had so queasy a feeling about my contribution that he did not fill the pages with learned red-ink disputation of the fine details of my proposals, as I expected, but simply said to me as he handed it back: "Like all psychologists, you have a scant regard for the facts!"

So I can sympathize with Shastri & Ajjanagadde (S&A). Here they are, responding, perfectly understandably, to a suprathreshold signal above the noisy hubbub of neurobiology, when this neurobiologist has to say that the phenomenon on which they base their ideas probably does not exist in the primate. As the primate brain is the only one that we know is capable of systematic reasoning, this may be a problem for them. But I sympathize even more with S&A, since only rather close inspection of the neuroscience literature reveals this problem: The oscillophiles cited in the target article – in what one hopes is just a temporary failure of elementary scholarship – never remark in print on work which disagrees with them. And the problems to which they studiously fail to refer are serious enough that they are not adequately dealt with in a few dismissive remarks in S&A's note 27.

Basically, the problem is that the cat findings do not replicate in primate visual cortex. For an alternative reading list on this, S&A might have considered Bair et al. (1992), Kiper et al. (1991), Gawne et al. (1991), Young et al. (1991; 1992), as well as the two papers noted in note 27, which they dismiss so lightly (Rolls 1991; Tovee & Rolls 1992). As far as I know, the Young et al. (1992) study is the most comprehensive primate study to date, so I will briefly review its contents.

We sought to replicate the cat findings in the monkey. To do this, we recorded multiunit activity (MUA) and local field potentials (LFP) in areas V1 and MT, and MUA from the inferotemporal cortex (IT) of macaques. Recordings in all areas were made under conditions of stimulation and anaesthesia as close as possible to those in the cat. In addition, we recorded MUA in the IT of awake behaving monkeys while the monkeys performed a face discrimination task. The data were analyzed with methods taken from Engel et al. (1990), so that the primate and cat results could be compared directly.

In V1, with drifting bar stimuli, all frequency spectra of the LFPs showed the greater part of their power to be concentrated in the low-frequency components, and on stimulation LFP power spectra showed broad band increases in amplitude and not a shift in power from low to mid-frequency as has been reported in the cat. Indeed, the effects were almost the opposite of those in the cat: Stimulation was associated with statistically significant increases in power particularly at the low frequencies with a smaller increase across almost the entire spectrum. This wide-band stimulus-related increase in spectral power may have simply reflected that cells near the electrode fired more strongly when stimulated than when not, and did so at a variety of frequencies. The changes in frequency distributions would not provide narrow-band high-amplitude field potentials to which spike activity could become synchronized, and concomitantly, the oscillating MUA responses that we did see were in the alpha range, and there was no stimulus dependence.

In area MT, all LFP frequency spectra again showed most power to be concentrated in the low-frequency components and there were broad-band increases in amplitude on stimulation. All oscillating responses were in the alpha range and there was little evidence that they were stimulus related. In both V1 and MT, therefore, with *moving stimuli* (cf. S&A, note 27) that were very similar to those used in the cat experiments, there was no sign of the cat oscillation phenomena. It may be worth noting that this was not a "finding of no effect," which would not distinguish between insensitivity of the statistical analysis and the absence of the phenomenon: The stimulus effect in the LFP analyses was statistically reliable in all frequency bands except those centred on the alpha range, and the statistical procedures were able to detect oscillations at frequencies different from those observed in the cat.

In the IT of anaesthetized monkeys, no MUA responses indicated the occurrence of oscillation. In monkeys trained to make a differential response to a small set of human faces versus a larger set of faces (at which discrimination they achieved better than 90% correct performance), only two MUA recordings showed oscillations in the gamma range. One oscillating response was associated with stimulation and the other was associated with the absence of stimulation.

These results suggest that oscillating responses in the gamma-frequency band are remarkably rare in conditions very close to those in the cat studies and even in conditions that would be thought to require the binding of features into a representation coherent enough to form the basis on which the discrimination decision could be made. The fact that such oscillations were not stimulus dependent also suggests that oscillations are not required for feature binding in the studied regions of the monkey visual system.

Having used the methods of Engel et al. (1990) to classify the data as oscillating or not, we noticed a number of methodological problems with this and related methods. For example, the cat researchers forgot to take account of the goodness-of-fit between the Gabor functions (whose parameters were used to classify the responses) and the correlograms. Obviously, if one does not care how well a description fits, then Mrs. Thatcher could, for example, be described as an enthusiastic European: If the parameters of a description are to be used to classify something, the description should fit the described thing well. We found that Type I error due to this factor alone could vary between 17% and 100% overestimation. Similarly, "burst," "delayed inhibition," and "return" components, which are sometimes seen in correlograms and which could be consistent with being in the chaotic domain and not only the oscillatory domain, would unfortunately have been included as "oscillating responses" according to the methods of Engel et al. (1990). These methodological difficulties leave the empirical status of some findings in this area rather uncertain. For example, how can we know that the "long bar experiment" did not involve false positives?

It seems unlikely, in the light of these empirical facts, that stimulus-related oscillations could be a general phenomenon, and unlikely, therefore, that a periodic temporal "code" is a general solution to the problem of binding the separate features of an object, visual or semantic, into a coherent representation. This is a nice illustration of the dangers of having one's psychological theory disproved "by some irrelevant physiological research" (Broadbent 1958), and I suppose that S&A have two options. Either they take Broadbent seriously, and stop building models based, even loosely, on what's happening in neurobiology, which would be a shame, or they pay a bit closer attention next time.

As for the rest of S&A's target article, it seems to me to be terribly brave, but, in the end, just cognitive science.

Authors' Response

A step toward modeling reflexive reasoning

Lokendra Shastri[a] and Venkat Ajjanagadde[b]
[a]Computer and Information Science Department, University of Pennsylvania, Philadelphia, PA 19104 and [b]Wilhelm-Schickard-Institut, University of Tuebingen, Sand 13, W-7400 Tuebingen, Germany
Electronic mail: *[a]shastri@central.cis.upenn.edu; [b]nnsaj01@mailserv.zdv.uni-tuebingen.de*

Our response is organized into five sections. In section R1 we respond to issues concerning the biological plausibility of our model. In section R2 we discuss questions about its cognitive/psychological significance. Several commentators pointed to alternative approaches to dynamic bindings and reflexive reasoning. We discuss these in section R3. In section R4 we respond to some comments about learning. The remaining issues are discussed in section R5. In what follows we refer to our model as SHRUTI.[1]

R1. Biological plausibility

Before responding to commentaries about the biological plausibility of SHRUTI let us repeat what we said in section 1.4 of the target article: "Neural plausibility is an important aspect of this work – we show that the proposed system can be realized by using neurally plausible nodes and mechanisms, and we investigate the consequences of choosing biologically motivated values of system parameters. Needless to say, what we describe is an idealized computational model, and it is not intended to be a blueprint of how the brain encodes an LTKB (long-term knowledge base) and performs reflexive reasoning." We would like to stress that SHRUTI is an idealized computational model, and when we claim that it is biologically plausible we mean that it is possible to realize its essential components – the behavior of various types of nodes and the functionality of the proposed network – using neural wetware.

R.1.1. Synchrony, oscillations and biological plausibility. One of the major issues raised in the commentary concerns the biological *reality* of oscillations. With varying degrees of emphasis **Young, Freeman, Tsuda,** and **Eckhorn** point out that periodic (oscillatory) activity does not occur in the brain. Freeman and Young even go on to suggest that SHRUTI is therefore not biologically plausible. This conclusion rests on a mistaken understanding of the role of oscillations in SHRUTI.

R1.1.1. Oscillations are not essential for the functioning of SHRUTI. The essential feature of neural activity required by SHRUTI is *synchronization of cell activity and the propagation of synchronous activity along connected cell-clusters.* Since transient oscillatory activity seemed like a natural way of realizing such a behavior, we adopted oscillations in our model, but oscillations per se are not essential for the functioning of SHRUTI. It is therefore incorrect to link its biological plausibility with the existence or nonexistence of oscillations in the brain. The crucial question is this: Is synchronous activity biologically real? The nonessential role of oscillations in our model is clearly recognized by **Rohwer** and **Strong** (and to some extent by **Thorpe**).[2] Strong also points to some architectural simplifications that might result from dropping the periodicity requirement (but see R1.4). **Eckhorn** and **Freeman** do discuss the possibility that dynamic bindings may be represented by aperiodic (nonrhythmic) synchronous activity in the brain, but they fail to recognize that such a representation is compatible with SHRUTI.

The primacy of synchronization in the representation and propagation of dynamic bindings is pointed out at several points in the target article, including the title. "We represent dynamic bindings between arguments and fillers by the *synchronous* firing of appropriate nodes" (sect. 3, para. 3; see also sect. 1.3, para. 3; and sect. 3.1, para. 2). The behavior of ρ-btu nodes in section 3.2 and τ-and nodes in section 3.3 is defined in terms of general synchronous activity and then elaborated for the case of oscillatory activity. The output of τ-or nodes has been specified as being oscillatory. This is not critical, however, and it is trivial to modify the design so that the output of τ-or nodes may be assumed to be a burst of activity whose duration is comparable to π_{max}.

In the aperiodic case, the parameters π_{min} and π_{max} correspond to the minimum and maximum allowable time between two consecutive firings of a cell-cluster involved in synchronous activity. The interpretation of ω continues to be the width of the window of synchrony (see sect. 3.1, last para.). So the basic architecture of SHRUTI remains the same even if we admit aperiodic synchronous activity. The propagation of bindings now parallels even more closely than before the propagation of activity along "synfire chains" (Abeles 1982).

It is important to note that dropping the requirement of periodicity does not change the predictions about reflexive reasoning. The restriction on the form of rules, the bounded depth of reasoning, and the constraints on the capacity of the WMRR (working memory underlying reflexive reasoning) remain the same. The exact numeric value of the ratio π_{max}/ω, however, may have to be revised using the appropriate neurophysiological data.

R1.1.2. What led us to oscillations? When he asserts that our ideas and SHRUTI are *based on* the phenomenon of oscillatory activity in the animal brain, **Young** (para. 2) has it backwards. On the contrary, the design of SHRUTI was driven by the computational constraints on connectionist models enumerated in section 1.2 and the complexity requirements of reflexive reasoning discussed in section 1.1. The computational constraints prompted us to use temporal synchrony as a basis for representing dynamic bindings (to obviate the need of propagating pointers or symbols). We chose periodic (oscillatory) activity simply because oscillations seemed like the most straightforward and natural way of incorporating synchronous activity. As **Thorpe** points out, it was only later that we heard about the evidence for oscillatory activity in the brain.

R1.1.3. Can we conclude that oscillations are ethereal? The biological reality of oscillations is a matter of controversy. There are a growing number of reports of oscillatory activity in the brain – these include findings in the cat (see target article for references), squirrel monkey (Livingstone 1991), macaque (Engel et al. 1992; Kreiter et al. 1992), and even humans (Lado et al. 1992).[3] At the same time, we have negative evidence concerning oscillatory activity – we cite some papers in the target article and **Young** cites some additional findings. So the issue is far from settled. In spite of such conflicting evidence, Young's *emphatic* assertion that oscillations do not occur in the primate brain and are unlikely to play a role in the representation of dynamic bindings does not seem justified.

R1.2. Expected nature of oscillatory activity. Let us assume that oscillatory activity underlies the representation of dynamic bindings during reflexive reasoning. What sort of activity should we then expect to find in the brain during an episode of reasoning? The answer would vary dramatically depending upon our expectations about the nature of representations used by the brain. It is crucial that we recognize this, because not doing so may lead to erroneous expectations about the nature of oscillatory activity in the brain, and in turn, to wrong interpretations of raw data. We address this question in the context of periodic activity but our remarks also apply to aperiodic activity.

If one believes in fully distributed representations and assumes that entities are represented as patterns of activity over large populations of cells, one would expect a large number of cells to participate in oscillatory activity during an episode of reflexive reasoning. On the other hand, if one believes in more compact representations of the type adopted in SHRUTI, one should only expect a *relatively* small number of cells to participate in oscillatory activity.

Now consider **Freeman's** observation (para. 3) that periodically firing cells form a small tail in a distribution of firing rates and that a majority of cells yield a pulse interval that is more Poisson than periodic. What one concludes from the data would depend on one's assumptions about the nature of representations. Someone who believes in distributed representation would be compelled to conclude that oscillatory activity does not underlie the representation of dynamic bindings, but someone who believes in more compact representations would find good evidence for the hypothesis that oscillatory activity underlies dynamic bindings; because only a very small fraction of cells would be involved in oscillatory activity at any time, the small tail constitutes just the right evidence!

R1.2.1. The nature of oscillations predicted by SHRUTI. Let us consider a thought experiment to illustrate the nature of oscillatory activity entailed by a SHRUTI-like system. Assume that the system is in a "quiescent" state, namely, it is not receiving any stimulus and is not engaged in any systematic thought. At this time the nodes in the system would be firing with some background rate, perhaps Poisson. Now assume that the dynamic fact "John bought a Rolls Royce" is injected into the system. We would expect two resultant trains of oscillatory activity to propagate in the system. One would originate at the John and buyer clusters and rapidly expand to include other clusters representing owner, person, wealthy, and so on. A second train of activity would originate at the Rolls-Royce and buy-object clusters and expand to include other clusters such as car and own-object. This oscillatory activity might last only a few milliseconds, after which the synchronization would probably break down. The active nodes may, however, continue to fire at a high rate for some time before reverting to the background rate of firing.

The model posits that arguments and focal nodes of concepts are encoded by small clusters of cells. Even if the reflexive reasoning following an input results in the activation of several hundred relations (predicates) and types/features, the total number of nodes engaged in synchronous activity during an episode of reasoning may remain small – perhaps about 10^5 cells. Furthermore, this activity would be distributed across the area(s) where conceptual knowledge is represented. This estimate is extremely crude and speculative and may be off by an order of magnitude, but it still conveys the essential point: A very small fraction of cells (perhaps as few as one in about a hundred thousand) may be involved in synchronous activity during an episode of reasoning (this already assumes that we are focusing on some appropriate 1-10% of the brain where we expect conceptual knowledge to be represented).

R1.2.2. A fully developed SHRUTI-like system will have complex dynamics. Consider a fully developed SHRUTI-like system incorporating the functionality outlined in sections 10.1-10.4. The extended system would be capable of responding to continuous stimuli and of shifting its focus of attention. The dynamics of such a system would be far more complex than the simple oscillatory patterns depicted in the examples shown in the target article. The frequency of its oscillations would vary constantly because frequency increases whenever entities "leave" the WMRR and decreases whenever other entities "enter." Different modules in the system would be firing at different frequencies and each would have its own phase distribution. It is not at all surprising that the oscillatory activity observed in the brain is far more complex than the activity portrayed in the target article! We had noted this in section 7.1.

R1.2.3. Chaos and oscillations. An alternative, much more complex description of neural activity based on the notion of chaotic oscillations is offered by **Tsuda.** The relation between his characterization and SHRUTI needs to be examined further, but it appears that Tsuda may be underestimating the degree of systematicity required for supporting reasoning of the sort we discuss in the target article. The dynamics he describes seem more apt for a system that is not engaged in systematic reasoning resulting from a specific stimulus, and the activity of a SHRUTI-like system in a disengaged state could well be chaotic, but we think the activity of the appropriate subset of nodes would have to get organized rapidly once the system engages in systematic reasoning.

R1.3. Dynamic bindings and neural communication. Section 3 of **Eckhorn**'s commentary suggests that he may not realize how much information must be transmitted to communicate dynamic bindings during reflexive reasoning. In discussing the limited ability of a neuron to transmit symbolic information we had estimated the amount of information transmitted in 15 msec to be about 2 bits (see Note 4). This was based on the assumption that the firing rate typically varies between 1-200 spikes/sec. Eckhorn, however, argues that the maximum rate of firing can be *as high as* 300 bit/sec, and that if we assume a 20 msec cycle time, the amount of information transmitted by a neuron can be as high as 8 bits. Unfortunately, this does not change the situation one bit! Neither 2 nor 8 bits are sufficient for solving the dynamic binding problem during reflexive reasoning. The number of bits a neuron would need to transmit to communicate the identity of an argument filler will be more than 20. Contrary to what Eckhorn seems to imply, the number of distinct entities that may fill arguments in dynamic bindings is not 500 but closer to 100,000.[4] This means that even if we were to assume *perfect* coding and *noiseless* communication we would require 20 bits to communicate the identity of each filler.

Eckhorn also suggests that the neuronal limitations in communicating symbolic information may be overcome by using clusters of neurons rather than single ones. In section 9.4 of the target article, we discuss such a possibility and point out the advantages of using the temporal synchrony approach.

In paragraph 5 of his commentary, **Thorpe** suggests there is a tension between the fact that SHRUTI takes several cycles of synchronous activity to compute a response and other evidence suggesting that neurons respond within just one or two spikes (cycles). He seems to be overlooking the fact that an episode of reasoning takes *several* cycles of synchronous activity because it involves the propagation of synchronous activity over several layers of cells – as many layers as the length of the chain of reasoning. The propagation of activity across *each* layer, however, only takes 1-2 cycles (spikes). This is exactly what one would expect in view of Thorpe's discussion in his commentary (paras. 3, 4).

R1.4. Complexity of node types and circuits. Several commentators (**Dawson & Berkeley, Diederich,** and **Garson**) suggest that the node types used in SHRUTI are not biologically plausible. The behavior of ρ-btu nodes is eminently plausible and if Abeles (1982) is right about the significance of synchronous activity and synfire chains, it can even be argued that a ρ-btu node with an appropriately high threshold is a reasonable idealization of a neuron. The other two types of nodes, namely, the τ-and and τ-or nodes, are best viewed functionally as simple circuits made up of a small number of cells.

Cottrell, Dawson & Berkeley, Diederich, Garson, Koerner, and **Touretzky & Fahlman** remark that some of the circuits used in SHRUTI, particularly the multiple instantiation switch networks, are too complex and specific to be biologically plausible. First, the switches described in the target article are intended to demonstrate that it is possible to achieve the desired control and functionality by circuits made up of ρ-btu, τ-and, and τ-or nodes. These circuits amply demonstrate this possibility. Second, we agree that these circuits are quite specific and complex. To put things in perspective, however, we would like to point out that the concern about the circuits' being too specific and complex to be biologically plausible is misplaced and stems in part from the tacit assumption that these circuits have to be learned by an agent. There is no reason, however, to assume that such circuits – or, rather, circuits that are functionally equivalent to these – are learned developmentally. It is enough to assume that they have been "designed" by a process that operates at the evolutionary scale. Surely evolutionary processes are capable of crafting something as *simple* as the circuit in Figure 22. To think otherwise amounts to ignoring the intricacy, specificity, and complexity of the brain, not to mention the human body. Note that the internal circuitry of each switch is the same, so the same circuitry can be replicated over and over again. Learning need only involve connecting the input and output "wires" of preexisting switches to the input and output wires of concept clusters.

On a different note, **Strong** suggests that the design of a concept cluster may be simplified if the periodicity requirement is relaxed. This proposal is interesting because it allows the potential of sharing nodes and seems to be capable of self-induced phase separation. Although we can see how the proposed alternative allows multiple instances of a concept to be represented, it is not clear how it solves the difficult technical problem of communication between two concept clusters. It would be instructive to generalize the proposal to encode *n*-ary predicates so that several predicate instantiations may be represented without cross-talk. It would also be interesting to see how the arguments of antecedent and consequent predicates can be linked to ensure that bindings pertaining to several instantiations may propagate without cross-talk.

R1.5. Timing estimates. It is argued by **Garson** and by **Hirst & Wu** that the nodes in SHRUTI do not correspond to actual neurons and that it is therefore inappropriate to conclude anything about the actual time course of reflexive processing based on an analysis of our model. The timing data we present in section 8.1.1 are meant to be a broad indicator of reasoning times and their main purpose is to *demonstrate* that reflexive reasoning can be performed within a few hundred milliseconds by a system of simple and slow computing elements. Note that the basis of our estimates is the time it takes synchronous activity to propagate from one cluster of cells to another. Our estimates are therefore not too sensitive to actual encoding details as long as the number of layers in an alternative implementation of the switches and clusters is comparable to the number of layers in our implementation.

R1.6. Restriction on the number of entities. Our estimates of the maximum number of different entities that can be referenced in the WMRR are challenged by **Koerner.** He argues that we should not use data about the periodicity of oscillations of cells involved in early visual processing to make inferences about complex psychological processes, citing his own interpretation of the 7 ± 2 limit based on the much slower θ activity.

Our estimates apply to first-order bindings, those for which argument fillers are entities and not dynamic relational structures (an entity may be a complex relational structure as long as it is a static one). We believe that reflexive reasoning primarily involves first-order bindings; for such bindings, the appropriate interpretation of π is the cycle time of relatively *small* cell-clusters participating in synchronous activity (periodic or aperiodic). This interpretation of π would, we believe, apply to first-order bindings involved in reflexive reasoning as well as vision, modulo variations in the characteristics of different cell types.

A possible reason for **Koerner**'s objection may be that he is referring to higher-order relational structures such as patterns of activity corresponding to *complete solutions* and *hypotheses*, in which fillers themselves can be dynamic relational structures. If, as Koerner suggests, temporal synchrony is used to prevent cross-talk among such complex and large dynamic structures, the associated temporal activity is likely to have a much larger cycle time and hence a much higher value of π.

R1.7. "Local" representations and biological plausibility. It is suggested by **Touretzky & Fahlman** (paras. 5, 6) and by **Hirst & Wu** that SHRUTI is not biologically plausible because it uses localist representations! The representation of a predicate in SHRUTI is not a single node; a cluster is $n + 2$ nodes (n role nodes and 2 τ-and nodes). As **Hummel & Holyoak** point out, this means that our representation of a predicate instance is a distributed pattern (though not in a holographic sense). In addition, each argument node maps to several cells that may be physically distributed. So our model makes use of a physically "distributed" representation even though the representation of an argument is conceptually "localist." In view of the above, it is not clear which biological axiom Touretzky & Fahlman and Hirst & Wu think we are violating. If they are suggesting that our model does not adhere to the holographic version of distributed representation then we refer them to Hummel & Holyoak's commentary and section R3.1, where it is argued that such representations cannot support systematicity and knowledge-level parallelism.

R1.8. Are brain mechanisms totally distinct across modalities? "Whether or not the brain makes use of temporal synchrony in object perception has *no bearing* on how we reason abstractly," writes **Sloman** (para. 2, emphasis added). **Diederich** (para. 8) expresses a similar concern (though in a milder form). We think Sloman's stand is an extreme one. There are good reasons to suspect that the mechanisms developed for perceptual and motor processing were coopted by the brain to solve other cognitive problems.

R1.9. Central control and SHRUTI. Doubts are expressed by **Koerner** about our view that the reflexive-reasoning process can run without a central controller. He suggests that central control will be required during reasoning and decision making. He envisions such a controller guiding the activity into a "globally consistent" state and "focusing" the "search" toward additional support for the chosen hypothesis. We welcome his comments and pointers to his work on related problems, but we think they pertain more to reflective, than reflexive, processing. Notice that what he describes seems like deliberative reasoning and decision making where the system must choose one of several competing hypotheses, find a *globally* consistent state, and carry out *focused* search for support. This is exactly the sort of processing we have described as reflective (see sect. 1). It is quite likely that such reflective reasoning requires highly controlled and focused activity driven by an attentional mechanism (see sect. 8.2.2).[5]

Note also that the view of representation and processing put forth by **Koerner** relies much more on finely structured temporal activity than on the kind envisioned in SHRUTI (even in its extended form). One of the features of SHRUTI is that it uses a great deal of spatial structure to reduce its dependence on the fine-grained structure of temporal activity. It is possible that a greater dependence on fine-grained temporal structure has led Koerner to the conclusion that central control is necessary for controlling the temporal aspects of activity.

Dawson & Berkeley (para. 2) and **Touretzky & Fahlman** (paras. 8-10) confuse an "interface" with a "central controller," accordingly arguing that our system already has a (hidden) controller! We respond to this in section R5.3.

R1.10. Some imaginative arguments against the biological plausibility of SHRUTI. Our system is not biologically plausible **Dawson & Berkeley** (para. 5) claim, because it requires an "external learn signal" to trigger the learning of a fact. Their observation is based on a statement in section 10.5 about a scheme for storing facts in medium-term memory. We indicated there that a fact is learned in the presence of a "learn signal." Surely it is plausible that an *internally* generated signal based on the novelty or salience of an input allows the *one-shot* learning of a situation (fact). Next, Dawson & Berkeley invoke the biological implausibility of backpropagation to claim that our system is not biologically plausible. The most imaginative objection to the biological plausibility of our system, however, is that the limitation of the (logical) inferential power of our system "casts further doubt on its putative biological plausibility"! To suggest that SHRUTI is not biologically plausible because it lacks inferential power seems to miss the point altogether.

R2. Cognitive significance

The comments about the cognitive significance of SHRUTI cover a wide range. On the one hand, we have **Touretzky & Fahlman**'s sweeping dismissal of our model; on the other hand, we have **Oaksford & Malloch**'s enthusiastic appraisal of it as "potential landmark in the cognitive science/psychology of human reasoning"! We think an objective and careful evaluation of the target article will help the reader determine which of these is the better characterization of SHRUTI.

The criticism of SHRUTI centers around two themes: The first concerns the lack of empirical support. The second concerns incomplete coverage of the reflexive-reasoning phenomenon.

R2.1. Empirical support and cognitive modeling. We agree that, contrary to standard practice in psychology, we have not tried to replicate a specific data-set obtained in a laboratory experiment. We have pursued a very different approach, focusing on a set of space and time constraints obtained from some broad but well-grounded observations about the nature of reflexive reasoning (see sect. 1.1.1; see also **Hampson, Oaksford & Malloch,** and **Ohlsson**). At the same time, we have also focused on a set of computational constraints that characterize the architecture underlying cognition – in particular, its limited ability to process and communicate symbolic information (sect. 1.2). What is described in the target article is (1) a detailed model that satisfies these two sets of constraints and (2) the psychological implications of the model.

In suggesting that our model is rich in assumptions **Sloman** may be missing the significance of these computational and architectural constraints and labeling them as mere assumptions. We sympathize with him for this confusion. As **Oaksford & Malloch** point out (para. 3), our approach is not very common in cognitive psychology,[6] where issues of computational effectiveness and scalability are often secondary and the emphasis is on building empirically adequate models that fit some well-circumscribed body of data.

R2.1.1. SHRUTI and predictions. In addition to satisfying the general set of constraints on space-time resources and information processing abilities of nodes and links, SHRUTI also leads to several specific and testable predictions about the nature of reflexive reasoning. A number of psychologists have remarked on the significance of these predictions (see **Diederich, Hampson, Hummel & Holyoak, Oaksford & Malloch,** and **Ohlsson**). As described in section 8, SHRUTI predicts the capacity of the working memory underlying reflexive reasoning (WMRR) and the form of rules that can participate in such reasoning. SHRUTI also predicts that the maximum depth of derivations during systematic reasoning will be shallower than that of associative priming (sect. 8.2.6).

Oddly enough, **Sloman** and **Touretzky & Fahlman** discount all these predictions. Inexplicably, Sloman dismisses the predictions concerning the capacity of WMRR in suggesting that Baddeley's work "already accounts for working memory data"! Yet the target article explicitly states that the WMRR is the functional description of the dynamic activity of the LTKB and is quite distinct from the notion of working memory studied by Baddeley (see sect. 8.2.2). We would like to stress that our predictions about the capacity of WMRR are not only potentially important for reflexive reasoning, but they may also lead to insights into other reflexive phenomena as well (e.g., see Henderson's [1992] work on parsing).

Sloman also describes the restrictions on the form of rules identified in section 8.2.5 as "rather arbitrary" and "unlikely to be useful." He overlooks the discussion in section 4.9, where we suggested why the constraint may have a fundamental computational basis. Since the writing of the target article we have a proof that the constraint on the form of rules is an *essential* one for reflexive processing (Dietz et al. 1993). In other words, reasoning involving rules that violate the constraint in question cannot be carried out using space that is only linear in the size of LTKB and time that is independent of the size of LTKB.

Sloman goes on to argue that our predictions about the limitations on the use of transitivity are also wrong. His counterexample seems to be based on a misunderstanding of the issue at hand. It does not establish that people can compute transitivity with ease; it simply demonstrates the rather obvious fact that people are good at identifying certain linear orderings – in this case the natural ordering of integers! Note that an agent can determine whether *Pimplier(i,j)* simply by noting whether or not *i* is less than *j*!

Surely **Sloman** must realize that his example does not provide an appropriate test of the prediction. Consider the following "experimental data": After being shown the sequence Foo_1, Foo_2, . . . , Foo_{37}, subjects were able to recall correctly all the 37 items in the sequence, furthermore, they were also able to recall the exact order in which the items were presented. Would Sloman conclude from this that our memory span is 37 and that there is no recency effect?

As for explaining human fallacies, the model makes a number of predictions about when people might provide no answer or a wrong one. The constraints on the form of rules, the capacity of WMRR, and the depth of reasoning all point to the numerous ways in which human reflexive reasoning is fallible. We are also aware that we have not modeled all sources of error and all factors that lead to nonprescriptive behavior (e.g., see sect. 5, para. 2). The phenomenon that **Sloman** refers to, namely, the graded nature of category inclusion, is the type of phenomenon that is relatively easy to model in a connectionist network using weighted links, so we are not surprised that Sloman has a "simple model" that mimics this specific effect.

Touretzky & Fahlman set up a false contrast when they cite the work of Collins and Michalski (1989) in evaluating the significance of our work. The two efforts are motivated by very different concerns and goals. Collins and Michalski's work is clearly significant, but it does not address the problem of reflexive reasoning, computational effectiveness, or biological plausibility.

We do agree with **Sloman** that it would be nice to see how our model could comprehend a simple story. We also agree with **Ohlsson** that the results of the model need to be integrated with existing psychological theories and with **Martin** that it is important to identify the relevant empirical data that would serve to corroborate the reflexive/reflective distinction suggested by SHRUTI. **Oaksford & Malloch** (para. 7) point to experimental results that provide corroborative data about the reflexive/reflective distinction. We hope other cognitive psychologists will also contribute in this regard. In this context we would like to add that the validation of the constraints proposed by the model need not come from the area of reasoning alone. They may also be validated by examining their implications for parsing, another reflexive phenomenon, and one for which there exist extensive empirical data. Henderson's (1992) work on parsing, using a SHRUTI-like model, is beginning to show that such restrictions help explain some of the limitations of human parsing by modeling certain garden path phenomena and people's limited ability to deal with center-embedding.

R2.2. Questions about coverage: Red herrings and real issues. Several commentaries raise the issue of coverage, pointing out that SHRUTI does not model every type of reflexive-reasoning behavior. These include several insightful remarks by **Barnden** and **Hummel & Holyoak** and valid observations about the lack of a treatment of negation by **Cottrell** and **Garson. Munsat** and **Bauer**

seem to have arrived at a genuine misconception about the reasoning ability of a SHRUTI-like system, whereas **Touretzky & Fahlman** first caricature our model and then dismiss it as uninteresting!

R2.2.1. Logic, deduction and SHRUTI. According to **Touretzky & Fahlman,** our model is simply a limited theorem prover. They dismiss our work based on this claim, arguing that because human reasoning is not purely deductive, SHRUTI cannot be a credible model of it. Sound argument, but the premise is unfortunately false.

Hirst & Wu state that not all reasoning is deductive, enumerating five problems they (rightly) claim require nondeductive reasoning. We agree (see Ajjanagadde 1991; Shastri 1988a; 1988b).

Let us reiterate the basic technical problem we have solved: We have extended the representational adequacy and inferential power of neurally plausible models by demonstrating how connectionist networks can (1) represent relational structures in a dynamic manner and (2) propagate such structures efficiently and systematically in accordance with "rules." Note that the essential characteristic of a rule is that it specifies a systematic mapping between the roles of relational structures. As pointed out in the target article (1) these relational structures can be viewed as schemas or frames, hence rules may be thought of as mappings between schemas or frames, and (2) the rules (mappings) can be "deductive" or "evidential," in particular, they may be sensitive to the type/features of the role fillers in a given situation.

The representational significance of the mechanisms developed in SHRUTI extends beyond deductive reasoning; the ability to represent and systematically propagate relational structures dynamically lies at the core of not just deduction but also evidential, abductive, and analogical reasoning. We discussed this in section 3.5 and at several places in the target article, pointing out the broader representational significance of being able to deal with predicates, variables, rules, and dynamic bindings (see, e.g., sect. 1.3, last para.; sect. 2.5; sect. 10, para. 1).

In section 2 we made what we believe is an important distinction between systematicity and appropriateness that helps distinguish the problem of representing and propagating relational structures from the issue of the strength of such a propagation. In section 5 we showed how these two factors could be integrated by making the propagation of bindings from one structure to another sensitive to the types/features of the fillers in the source structure. The idea that the strength of a rule firing can be defined as a function of the types of argument fillers is an important one and does more than "lend a little more [flexibility to] *modus ponens.*"

It is true that we carried out a detailed treatment of deductive reasoning only in the target article. We did this to investigate fully the strengths and weaknesses of the temporal synchrony approach as a mechanism for representing and systematically propagating relational structures. Note, however, that the predictions about WMRR capacity and the restriction on the form of rules would apply not onl to deductive reasoning but also to evidential, abductive, and analogical reasoning. This should further convince the reader of the methodological significance of separating the issues of systematicity and appropriateness. Unlike **Touretzky & Fahlman** and **Hirst & Wu,** several other commentators (**Barnden, Feldman, Hampson, Martin, Oaksford & Malloch, Ohlsson, Palm,** and **Strong**) apparently had no difficulty seeing the broader significance of our model.

R2.2.2. Reflexive reasoning is limited reasoning. Not only do **Dawson & Berkeley** seem not to have understood what our work is about (they think that it is only relevant for deduction), but they also criticize it for not being as powerful as a full-blown theorem prover. Restating the constraints we identified on the capacity of WMRR and the form of rules, they write that "although these two limitations are acknowledged," we have failed to note "the full extent of the problems they produce." Dawson & Berkeley don't realize that we consider the identification of these "limitations" a major contribution of our work in that it helps delineate reflexive reasoning from reflective reasoning.

R2.3. Plausible and possible inference. We agree with **Bauer** that a reflexive reasoning system should be able to separate plausible inferences from the vast set of possible inferences and that a "model" should be capable of representing implicit information that can be made explicit if and when the need arises. Bauer seems to have the incorrect belief, however, that SHRUTI lacks these attributes. This may stem from wrongly assuming that (1) SHRUTI only encodes plausible inferences in the LTKB, and (2) everything SHRUTI infers has to be explicitly represented in it.

Possible inferences. Consider the case of forward (predictive) reasoning. Given an input, the set of inferences that SHRUTI can draw using its LTKB corresponds to the set of possible inferences. In the purely deductive case, this set consists of the inferences that can be derived by the repeated application of *modus ponens* to the rules in the LTKB plus the input, *without exceeding* the capacity limitations of WMRR and the bound on the length of individual derivations.[7] Notice that there is a clear distinction between (1) the set of all *possible* deductions that SHRUTI can compute from its LTKB plus the input, and (2) the set of all *logically* possible deductions that follow from the same LTKB plus the input. The former excludes a large number of valid deductions whose derivations would cause WMRR capacity to be exceeded or whose length would exceed the depth bound. Hence SHRUTI provides a natural explanation for why a large class of valid deductions cannot be made.

Plausible inferences. The possible inferences drawn by SHRUTI will soon decay because of a dispersion of synchronous activity unless they are reinforced by subsequent inferences (see sect. 8.5). Note that inferences that reinforce each other are the ones that produce the same dynamic bindings. This means that in a system based on temporal synchrony, inferences that reinforce one another produce coherent activity (literally) and therefore survive long enough to affect other processing or get stored in medium-term memory. Thus plausible inferences correspond to inferences that are reinforced by other inferences or inputs and therefore survive. Implausible inferences are the ones that stand alone, and hence, soon decay. SHRUTI predicts that after each input, all possible inferences get drawn but only the plausible ones survive.

Consider an extended system encoding evidential rules and the ability to combine forward and backward reasoning, in other words, a system capable of abductive reasoning. Now consider the input "John bought a book" followed by "It is Susan's birthday." The input "John bought a book" will lead to a number of *possible* inferences. These might include, among other things: "John wants to read the book" and "John wants to give the book to someone." The input "It is Susan's birthday" will likewise lead to number of possible inferences. Some of these will reinforce the inferences from the previous input. In particular, some of the inferences triggered by "It is Susan's birthday" will reinforce the prior inference "John wants to give the book to someone" and also provide the binding "Susan" for the recipient role. Thus, the inference "John wanted to give the book to Susan" may emerge as the coherent (plausible) inference and survive, whereas the inferences "John wants to read the book" may decay.

Another apparent misconception of **Bauer's** is that SHRUTI can only infer something that is represented explicitly in LTKB! This is not the case. SHRUTI is capable of performing *inference*, which means that it can make explicit things that are only implicit in the LTKB or the input. For example, if the system is told "Harry bought a Rolls-Royce" it infers "Harry owns a car" and thereby makes explicit something that was only implicit in the input.

The example Bauer gives (about driving to the store) can easily be accounted for in the system described in the target article. Bauer's example simply illustrates that one of the "rules" in the LTKB should embody the following commonsense knowledge:

If an agent goes from a source *s* to a destination *d* during a time interval $[t_1,t_2]$, then for any location *l* on the path from *s* to *d* there exists a time *t* in the interval $[t_1,t_2]$ such that the agent will be at *l* at time *t*.

A little pause will convince the reader that the knowledge expressed above would be part of our common sense. The LTKB can also be assumed to include other pieces of common sense such as "*driving* from *a* to *b* implies *going* from *a* to *b*" and "the distance from the source to a point along the path is a fraction of the total path distance." If the LTKB contains such commonsense knowledge, then given "John drove from his home to the store," a SHRUTI-like system will be able to answer all the queries of the form, "Did John drive a third of the distance between his home and the store?"

R2.4. SHRUTI and the LTKB assumption. Like **Bauer, Munsat** also expects the right sort of behavior from a reflexive reasoner. He feels that neither SHRUTI nor any other system based on the LTKB-assumption (see Munsat) can embody reflexive-reasoning ability. We think this is too pessimistic and that an extended SHRUTI-like system would be capable of performing the sort of reasoning described by Munsat. His misgivings arise partly from the same set of misunderstandings that led Bauer to conclude that (1) everything inferred by SHRUTI has to be represented explicitly in the LTKB, and (2) there is no distinction between possible and plausible inference.

It is surprising that **Munsat** wonders how the right rules and facts become active from among the millions of rules and facts, because this is one of the core problems SHRUTI addresses! Perhaps Munsat wrongly thinks the LTKB is an *unstructured* set of propositions. If the LTKB were an unstructured set of propositions, Munsat's concerns would certainly be appropriate. In SHRUTI, however, the rules in the LTKB are highly organized and form an inferential dependency graph in which they are direct (hardwired) mappings from predicates to predicates and provide the necessary inferential paths for the automatic and efficient computation of inferences. In the case of an input, these hardwired mappings lead to all and only the possible inferences that follow from the input. For example, the input "Sally bought a Rolls-Royce" would physically cause the activity "Sally owns a car" but not "the moon is a satellite."

Munsat also worries about the need for a homunculus to decide what makes sense. One of the appeals of connectionist models is that they offer an alternative interpretation of what it means to "make sense." These include (related) notions such as reaching a locally minimum energy state, being in an attractor state, or forming a stable coalition. These are related notions, and in the context of reasoning they correspond to activity states where, for example, the cause-and-effect relations between active predicates are mutually reinforcing.

Munsat rightly observes that people can tell you what would have to be the case for a story line to make sense. But isn't this exactly what abductive reasoning captures? So given "John slipped on the floor," an abductive reasoner might come up with the hypothesis "the floor might have been wet," and "someone might have mopped the floor." Of course, the LTKB would have to include the commonsense knowledge that the floor's being wet can lead to someone slipping and falling, and that mopping the floor causes it to be wet. But this is exactly the sort of commonsense knowledge we would expect to be in the LTKB of an agent.

About the joke from "Cheers." We believe that a reasonable modeling of the LTKB of an agent exposed to popular TV fare would allow the modeling of the joke in question. We do not think our ability to understand such jokes implies anything magical about the contents of our LTKB or our reasoning ability – at least not in any way that transcends our already remarkable ability to perform reflexive reasoning.

Garson rightly points out that it would be unrealistic to assume each predicate to have all the arguments required for accommodating the potentially large number of modifiers that might arise in various situations. The problem can be solved as follows: Predicates are assumed to "inherit" arguments in much the way that concepts inherit attribute values. For example, arguments such as *location* and *time-of-occurrence* may be associated with the general predicate *event* and not replicated in predicates corresponding to more specific types of events such as *sell*. When the sell predicate is instantiated, the appropriate rule (the one that encodes: "sell is an event") will lead to an instantiation of the predicate *event*. Once event is active, its arguments *location* and *time-of-occurrence* would become available and may be bound to the value of location or time-of-occurrence provided by modifiers.

R2.5. Some real limitations. Several commentators have pointed out some real limitations in the expressive power of SHRUTI vis-à-vis reflexive reasoning. Although some of these can be readily overcome, others would require significant effort.

Cottrell (para. 4) and **Garson** point out that the target article does not deal with negation. The abductive reasoning system described in Ajjanagadde (1991) suggests one way of handling it. Cottrell points out another promising alternative. Referring to some of his earlier work, Cottrell cautions us that the introduction of negation might slow down some of the computations. The difficulties Cottrell refers to, however, were partially due to his use of default logic (Reiter 1980) as a framework for modeling inheritance with exceptions. We have argued elsewhere (Shastri 1988a) that default logic is not the appropriate tool for modeling what is essentially a problem of evidential/probabilistic reasoning. We have also shown that an evidential treatment of this problem leads to a connectionist network that can compute inheritance with exceptions effectively in time that is just proportional to the depth of the inheritance hierarchy. The inclusion of negation would support the encoding of rules such as $A(x) \Rightarrow \neg B(x)$ and would in turn allow the system to draw inferences of the type "John is not taller than himself."

Barnden (para. 8), **Hummel & Holyoak** (para. 3), and **Garson** (para. 7) point to an important restriction on the representation of dynamic structures in SHRUTI. SHRUTI can only represent dynamic structures containing first-order bindings – namely, the fillers of arguments in a dynamic structure must be entities; they cannot themselves be dynamic structures. Note, however, that an entity can be a complex structure as long as this structure is static, that is, built out of hardwired nodes and links.

A possible way of expressing higher-order bindings is to use a richer temporal structure than the one used in SHRUTI. In such a scheme, first-order bindings would be represented by very fine synchronization using short cycle times and narrow windows of synchrony, while higher-order bindings would be represented by coarse synchronization using long cycle times and wider windows of synchrony. **Koerner** seems to be advocating such a multilevel temporal representation. The problem with this approach, however, is that it can lead to complex and potentially unstable activity (see Koerner).

We believe that reflexive reasoning primarily involves first-order bindings and many problems that seem to require higher-order bindings can be reformulated so as to require only first-order bindings. For example, consider the representation of the *nested* structure: *go(John, path(at(home)))*, which may be read as "John went on a path that led to his home." A dynamic representation of this structure might appear to require a third-order binding for the second argument of *go*. Such a nested structure, however, can be expressed as a dynamic structure involving only first-order bindings by assuming that an instantiation of *go* creates a flat dynamic structure via the "rule":

$$\forall\ x{:}thing,\ y{:}thing\ go(x,y) \Rightarrow \exists\ p{:}path,\ l{:}location$$
$$go'(x,p) \wedge to'(p,l) \wedge at'(l,y)[8]$$

which says that *go(x,y)* means that there exists a path *p* and a location *l*, such that *l* is "*at y*," *p* is the path to *l*, and *x* *goes* on *p*. (Additional rules involving the predicates *go'*, *to'*, and *at'* would specify the meanings of the predicates *go'*, *to'*, and *at'*.)

In the target article we had conjectured that our ability to incorporate novel, rulelike information during reflexive reasoning may be extremely limited.[9] Although **Barnden** does not reject this claim outright, he expresses some doubts and offers a counterexample. But Barnden's is not necessarily a counterexample. As stated in section 5, paragraph 7, the use of types allows certain rulelike information to be expressed as a "fact." Specifically, if the "rule" involves only *unary* predicates in the antecedent then it can be expressed as a fact. Thus, instead of expressing "Everyone at the party was a toothbrush salesperson" as a rule $\forall x$ *at-party(x)* $\Rightarrow$ *sells(x, Toothbrush)* one could express it as *sells(at-party, Toothbrush)*, where *at-party* just refers to the set of people who were at a particular party. Barnden's example, however, does highlight that a reflexive reasoner should be capable of referring to dynamic "sets" such as "the people at a specific party."

R2.6. Relation between reflexive and reflective reasoning. Several questions are posed by **Ohlsson** and **Martin** concerning the relation between reflexive and reflective reasoning; some answers are provided by **Oaksford & Malloch** and **Hampson.**

Shift from reflective to reflexive. As suggested in the target article, rules that participate in reflexive reasoning must be integrated into the LTKB by being embedded in the inferential dependency graph. This integration is expected to be a slow process requiring repeated experience or observation. **Hampson** (para. 2) offers some supporting evidence and points out that this is consistent with the general view that practice shifts processing in the direction of automaticity (also see **Strong**). However, SHRUTI also predicts that rules whose form violates the restriction stated in section 4.9 cannot become part of a reflexive process.

Reflexive and reflective reasoning are not disjoint processes that use disjoint representations and mechanisms. We think that reflexive reasoning is our primary and basic reasoning mechanism. Reflective reasoning involves a combination of reflexive reasoning and additional mechanisms and representations. These would include an attentional mechanism for "remembering" a small number of input or inferred facts temporarily.[10] In other words, we will require an overt-STM that might very well correspond to the usual notion of a working memory (Baddeley 1986).

Ohlsson wonders why agents use reflective reasoning if reflexive reasoning is so efficient. The answer is quite straightforward: Agents resort to reflective reasoning because they *must*. If the amount of dynamic memory required for solving a problem exceeds the WMRR capacity, if the depth of reasoning required to solve a problem exceeds the depth bound of reflexive reasoning, or if the form of rules required for reasoning violates the form constraint, the agent will have to resort to reflective reasoning and use conscious deliberation, props, and/or other external representations (see **Oaksford & Malloch**, paras. 6-7).

R3. Paradigmatic issues

R3.1. Distributed representations: The magical alternative. The magical powers of distributed representations are invoked by **Garson** to suggest that our work is misdirected. Yet none of the existing models based on distributed representations come anywhere close to demonstrating the expressiveness, inferential adequacy, and scalability of SHRUTI. We hope that the proponents of distributed representations will recognize that a distributed system – at least in its pristine form – *cannot* have the necessary combination of expressiveness, inferential adequacy, and scalability. As **Hummel & Holyoak** point out, there is a basic tradeoff between distributed representation, systematicity, and parallelism; no amount of handwaving can make this tradeoff disappear.

A system using a distributed representation for arguments and fillers can only represent one dynamic binding at a time. How does **Garson** expect such a system to perform rapid reasoning (or parsing) within the desired time scale? It should not come as a surprise that DCPS and TPPS, two systems based on distributed representations, were serial at the knowledge level and could only apply one rule at a time.

Rohwer recognizes the advantage of using temporal synchrony, suggesting that to utilize space in an optimal manner one should use temporal synchrony in combination with distributed representation. But by using only temporal synchrony and interleaved node activity, a distributed representation system can only represent a *small* number of dynamic bindings. Hence the "optimal" use of space will mean giving up the ability to represent a large number of dynamic bindings simultaneously and knowledge-level parallelism.

As **Hummel & Holyoak** point out, SHRUTI also uses "distributed representations." An *n*-ary predicate is represented by a collection of $n + 2$ nodes and hence a dynamic fact is a pattern of activity distributed over several nodes. SHRUTI does use a localist representation of arguments (note, however, that although each role is localized in the *abstract* representation, it is *physically* distributed, because it is represented by a cluster of cells). The (abstract) localization of roles is essential in any system that must represent a large number of dynamic bindings simultaneously. Indeed, it is their localization that enables SHRUTI to represent and propagate simultaneously a large number of dynamic bindings and to exhibit knowledge-level parallelism.

As far as entities are concerned, the encoding of an entity can be viewed as a distributed pattern over the collection of nodes that make up the type hierarchy. If one augments the representation of types (concepts) with attribute values (see Shastri 1988a; Shastri & Feldman 1986) then the "distributed" nature of the representation of each entity becomes even more apparent. Observe that the key to encoding similarity is the use of *shared* representation and it is this sharing that gives distributed representations the ability to capture similarity. The type hierarchy also leads to such a sharing of representation and, hence, allows SHRUTI to capture similarity.

We agree with **Dorffner** that a reflexive-reasoning system should have a more fluid and dynamic view of compositionality. It should be capable of zooming in and out over representations effortlessly at different levels of granularity and of interpreting a situation/input relative to its current goals (re: Dorffner's example of our interpretation shifting from "a blob," "a ladder," to "a chair on a table"). But we fail to see what this ability has to do with distributed representations per se. Dorffner also advocates a flexible interpretation of roles. In a sense, SHRUTI already exhibits such flexibility. Consider the activity resulting from the input "John owns a car." This input will result in the roles "owner" and "potential-seller" firing in synchrony with John. One can view this activity over the role nodes as the distributed pattern corresponding to the "soft" role being filled by John. Now imagine that there are certain types of objects, say *foo*, that can be owned but not sold. This knowledge would be encoded as the appropriate type restriction on the rule between own and can-sell. Now if we present the input "John owns a foo," the resulting activity will only involve the role "owner" firing in synchrony with John. Thus in this situation, John can be viewed as filling a different role given by a different pattern of activity over the role nodes.

Halford offers a comparison of the tensor product approach (TPA) to dynamic bindings and the approach used in SHRUTI. We welcome the comparison but disagree with some of the specifics. For example, Halford says that the TPA representation of $R(a,b,c)$ also represents the influence of c on $R(a,b)$. But even SHRUTI's representation has a similar ability. Consider the representation of *give(John, Mary, x)* and the inferences that would follow from this partially instantiated relation. Now imagine introducing the binding (*g-obj* = *a-valentine*) in the above relation instance resulting in *give(John, Mary, a-valentine)*. A number of additional inferences would now follow. Would these additional inferences not denote the effect of adding "a-valentine" to *give(John, Mary, x)*? Halford also suggests that given $R(a,b,c)$, it is meaningful to talk about $R(a,b)$ in TPA. But does the ability to deal with partially instantiated relations not confer the same power on SHRUTI? Finally, Halford indicates that TPA supports the retrieval of any argument filler given the predicate and the remaining argument fillers. This seems to correspond to SHRUTI's ability to answer *wh*-queries (see sect. 4.7).

R3.2. SHRUTI and the classical approach. Several commentators see the ghost of classical AI in our model (**Dawson & Berkeley, Dorffner,** and **Garson**). Dawson & Berkeley also see SHRUTI as a mere implementation of classical ideas. Our response has two parts. First, we believe that any model of cognition will have to exhibit some of the functionality identified by the classical approach. We cannot simply discard the notions of systematicity and compositionality – what we need to do instead is discard the view that systematicity and compositionality have to be retained in their unconstrained and unfettered form. The interesting challenge is to determine the appropriate form and extent of systematicity and compositionality that cognitive models must support. If we draw the line too far to the left, we can end up with both a type of associationist glob that opponents of connectionism love to attack or models that work on toy examples but do not seem to have any hope of scaling to larger problems. If we draw the boundary too far to the right we can end up with attempts at building "connectionist" machines for doing list processing – an interesting exercise, but lacking any cognitive significance (Touretzky 1990).[11]

As **Feldman, Ohlsson, Martin, Strong,** and **Oaksford & Malloch** point out, ours is a different approach. We are trying to build a model of reflexive reasoning that respects the essential constraints imposed by the underlying computational architecture but that at the same time has (1) considerable representational and inferential power, (2) a limited yet potentially adequate ability to deal with systematicity and compositionality, and (3) requisite scaling power.

The comment that SHRUTI is a mere implementation of the classical approach simply misses the point. **Ohlsson** (para. 8), **Martin** (paras. 9-10), and **Oaksford & Malloch** (paras. 1-4) spell out the relation between SHRUTI and the classical approach. The key observation is that our "implementation" leads to a set of constraints and predictions about the nature of reflexive processing that are unique to this implementation.

Dawson & Berkeley offer three reasons why our model is a classical one. We have already responded to their comments about biological plausibility in R1.10 and we will respond to their claim that our system has a central controller in R5.3. Let us briefly comment on the "squiggly line" Dawson & Berkeley point to as the "smoking gun" that proves our system is a classical rule-based one! The squiggly line in Figure 19 was drawn to help the reader delineate the representations introduced in sections 3-4 from those introduced in section 5. By no stretch of imagination does this line separate the "data structures being processed" from "the rules governing system inferences." Both the type hierarchy and the rule base embody some data and some processing in the traditional sense of these terms. We strongly urge Dawson & Berkeley to reread section 3.4, paragraph 5, and the article by Hatfield (1991) cited therein.

R3.3. On the AI paradox. Our claim that we have taken a step toward resolving the AI paradox (see Abstract) is contested by **Hölldobler,** yet nothing in his commentary contradicts the basis for our claim, namely, that work in AI has not offered a credible account of how humans can rapidly perform a wide range of reasoning in time that does not seem to increase with the size of their knowledge base. The results in AI have either been negative and shown that even very "simple" types of reasoning are intractable, or they have offered characterizations of "complex" reasoning classes that require too much space or time, or produced positive results that are about overly restrictive forms of reasoning (see sect. 9 for references). So if our predictions about reflexive reasoning were to hold, we would indeed have taken a step toward resolving the AI paradox by showing that there exists a class of reasoning that can be performed with requisite efficiency and that is powerful enough to cover a significant range of reasoning that people can perform reflexively.

Hölldobler graciously observes that our "logic" has some remarkable features but he remarks that its expressive power is "fairly limited" from a "logical point of view." He does not seem to realize that the fundamental issue is not how powerful or weak reflexive reasoning is from a logical point of view. If it turns out that such reasoning corresponds to a "simple" logic, so be it! Hölldobler seems to want to hold us responsible for AI researchers' failure to investigate "simpler" logics.

We appreciate **Hölldobler**'s pointers to related work on automated theorem proving. Note, however, that the result Hölldobler discusses involves a stronger restriction on the form of rules than what we impose in section 4.9. Our restriction concerns only variables that occur *more than once* in the antecedent. Hölldobler mentions a stronger restriction that covers *all* variables occurring in the antecedent.

We are not surprised by Hölldobler's observation that as far as deductive reasoning is concerned, SHRUTI's inferential power is a special case of some more general-purpose theorem prover. This observation is of some value, but such a posteriori analysis should not be mistaken for the actual *identification* of an interesting and significant special case of a general problem.

R3.4. Do static bindings suffice? It is argued by **Cooper** that dynamic bindings may not be relevant because rule-based reasoning may be the wrong paradigm for modeling intelligence. As pointed out in the target article and in R2.2, however, the dynamic-binding problem transcends a narrow reading of "rule-based reasoning." Cooper refers to case-based reasoning (CBR) and implies that CBR may not require dynamic bindings! A little reflection, however, should make it clear that using a case would require binding (on the fly) its roles/slots to the appropriate entities in the current situation. Furthermore, any but a trivial CBR system would have to propagate some of these bindings in order to solve the indexing problem. Cooper argues that a full-fledged treatment of *n*-ary predicates may be unnecessary and counterproductive. We agree! Indeed, SHRUTI does not offer such a universal treatment (see sects. 4.9, 6, and 8).

Cooper also suggests that we need only solve the binding problem for *feasible* pairing of roles and fillers; because the number of feasible role-filler pairing is not astronomical, it may be possible to dedicate nodes to each of the feasible bindings. He concludes accordingly that it may be possible to get by without dynamic bindings. He seems to be making a crucial error, however, because a system must not only deal with feasible but also nonfeasible ones. Consider the sentence "The Grand Canyon gave a computer to a monkey!" Surely the bindings between *giver* and Grand Canyon are not feasible in Cooper's sense of the word. Yet we have no trouble in creating this binding and answering questions about who gave what to whom. So we are quite capable of representing essentially arbitrary pairings between concept/instances and conceptual roles without requiring repetition, attention, and reflection.

R4. Learning

A number of commentators point out that we have not investigated learning in detail. We agree. We also agree with **Martin** that pursuing learning within SHRUTI will provide an additional set of constraints that may lead to further insights into the nature of reflexive reasoning. In the target article we mentioned that we have a plausible solution to the problem of one-shot learning of facts. We are also pursuing the problem of incremental rule learning (see below). As **Hampson** points out, the integration of rules into the LTKB can be a slow and gradual process requiring considerable exposure to a variety of relevant situations. This is to be expected, because to learn a rule involves learning the correct argument mapping as well as the associated σ functions (see sect. 5.5) to embody the appropriateness of these mappings (**Cottrell** aptly refers to the σ functions as "semantic filters").

Grossberg summarizes work done by him and his colleagues on a family of learning systems that can extract rules from data. We would like to evaluate the power of these systems in the context of a SHRUTI-like reflexive-reasoning system.

Cottrell argues that using the pattern containing inference alternative (PCIA) discussed in section 9.4 will lead to a number of advantages in learning rules and semantic filters associated with rules. He does not realize that the temporal synchrony approach used in SHRUTI can support all the advantages he cites. This is illustrated in Figure R1, which makes it clear that semantic filters as well as covariance constraints can be learned within the temporal synchrony approach (a more detailed description of this approach will appear in Shastri 1993b). Figure R1a shows three groups of nodes. The one on the bottom left is the collection of all the predicate nodes. For simplicity, we have only shown role (argument) nodes of predicates and omitted the enabler and collector nodes. The group on the right consists of all the type or feature nodes. As discussed in R3.1, the representation of each entity can be viewed as a pattern of activity over the collection of type or feature nodes. The collection of nodes on the top is the "hidden" structure consisting of a layer of τ-or nodes sandwiched between two layers of ρ-btu nodes. The arrows indicate that nature of connectivity. Notice that the role and feature nodes feed into the bottom layer of the hidden structure and the top layer of the hidden structure feeds back into the role nodes. The interconnection pattern in the hidden structure is as shown: the bottom layer feeds into the second and third layer and the second layer feeds into the third layer. The proposed connectivity can be shown to support the learning of type/feature preferences/restriction involving individual roles as well as multiple roles.

Figure R1b shows the input activity for a particular situation (John walked into the wall) and the associated target activity (John got hurt). The role nodes will be

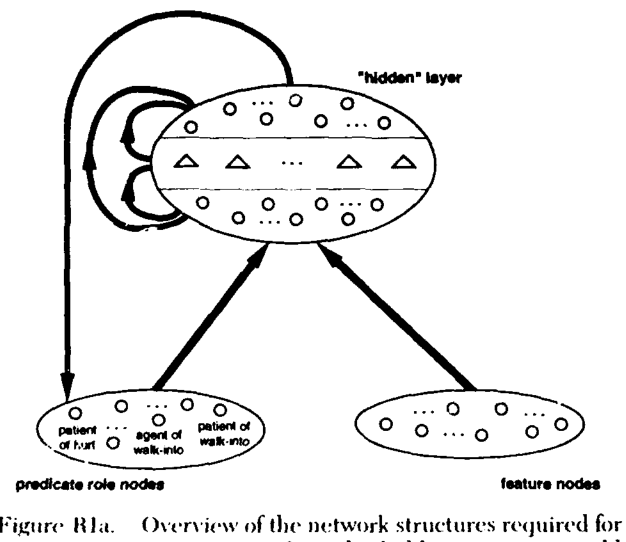


Figure R1a. Overview of the network structures required for learning context-sensitive rules. The hidden structure would develop the appropriate semantic filter for the propagation of bindings between predicate roles. Interconnections between collections of cells are indicated by the dark arrows. All links are weighted and nodes have thresholds associated with them.

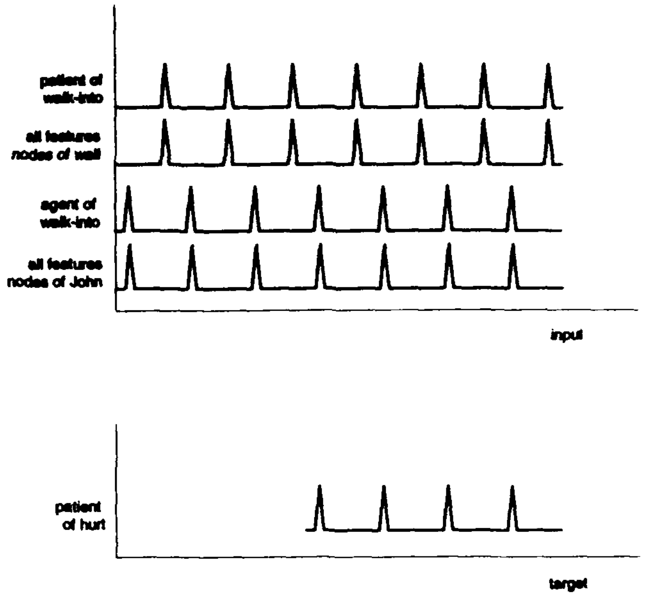


Figure R1b. The input and target patterns for the training situation: "John walked into a wall" (antecedent), "John got hurt" (consequent).

clamped to the input pattern shown in the figure (the input need not be periodic) and the desired behavior of the network would be the (suitably delayed) activation of the patient role of hurt as specified in the target pattern. One could use a suitable learning algorithm to learn the correct weights in the network to encode the necessary rules with the appropriate semantic filters. The above interconnection pattern should also make it clear that contrary to **Cottrell**'s (para. 7) and **Garson**'s (para. 6) suggestions, the learning of rules does not require one-to-one connectivity between all predicate role nodes.

R5. Miscellaneous issues

R5.1. Grounding. As pointed out by **Diederich**, we have not addressed how the meaning of our representation is ultimately grounded (Harnad 1990). We recognize that grounding is central to the notion of representation, but our concern in this target article has been with issues such as expressive power, inferential adequacy, and scalability of a biologically plausible representation and reasoning system. We will have to face the issue of grounding if we want to start ascribing *real* (not imputed) meaning to nodes and circuits in SHRUTI.

R5.2. Encoding of long-term facts and the *IS-A* hierarchy. In paragraph 2 **Strong** argues that the encoding of a partially instantiated fact like *give(John, Susan, x)* violates the closed word assumption (CWA). He writes that given the fact *give(John, Susan, x)* the CWA implies the answer to the question *give(John, Susan, Car7)* should be no. We agree; indeed, this is exactly how the system responds. So we do not see why he believes that SHRUTI's response *is inconsistent with the CWA. Strong is right,* however, about the encoding of a partially instantiated fact using the *IS-A* hierarchy. The reasoning system augmented with the *IS-A* hierarchy encodes a fact such as $\exists x$:*Thing give(John, Susan, x)* exactly as he describes.

Palm seems to be confusing long-term facts with medium-term facts. The intended function of what we have called "long-term facts" is indeed the essentially permanent recording of a situation (a static set of bindings). Some of this confusion can be resolved by recognizing that (1) only situations that are significant end up being encoded as long-term facts and (2) long-term facts are not entirely "forgotten" when the situation they encode ceases to be true. Instead, they are tagged in some manner to indicate that they are no longer the case. To draw an analogy, when long-term facts cease to be true, they do not just disappear; they continue to be around as "ex-long-term facts."

Palm seems to have misinterpreted the encoding of long-term facts in Figure 12, inferring that the enabler, collector, and argument nodes are duplicated for each long-term fact. This is not the case. As we explained in section 3.3, for each *n*-ary predicate there is only *one* enabler, *one* collector, and *n* argument nodes. All the long-term facts pertaining to this predicate share these "general" nodes. So in Figure 12 if we were to add the long-term fact *buy(Jackie, Car7)* we would only add one additional node, namely, a fact node.

The above confusion also leads **Palm** to think that multiple predicate banks introduced in section 6 are required for storing multiple long-term facts (end of para. 8). This is not the case. It turns out that multiple predicate banks have been posited for representing multiple *dynamic* instantiations of a predicate, not multiple long-term facts.

The suggestions by **Palm** about encoding *IS-A* and predicate hierarchies via set containment are well taken. We see two potential problems with his proposal: (1) representation of multiple dynamic predicate instances and (2) encoding of exceptional properties/features of concepts. Palm also comments about encoding soft rules and the potential problem with using rate of firing to encode confidence. He seems to have overlooked the discussion in section 5.5 and note 26.

R5.3. SHRUTI and a central controller. Overhasty dismissiveness seems to have led **Dawson & Berkeley** and **Touretzky & Fahlman** to confuse a simple means of communicating a query to SHRUTI and recovering the answer with a central controller (see paras. 2 and 10, respectively). They do not seem to understand that unless we develop a complete system which accepts sensory (speech/visual) input and produces [illegible]ch/motor output, we need to specify a way of co[illegible]ating with the system. We fail to see how posing [illegible] to SHRUTI by activating the argument and filler nodes and the *enabler* of the query predicate and thereafter waiting for the *collector* node of the query predicate to become active can be confused with the NETL-like requirement that a central controller direct the activity of each node at each step of the computation!

Touretzky & Fahlman misconstrue our reasonable conjecture that for linguistic input the phase separation in the activity of distinct entities might begin during the parsing process, which was made in the context of discussing how the activity in a system might self-organize so that each distinct entity starts firing in a distinct phase. Our conjecture is interpreted as our somehow hiding the central controller in a parser! They also fail to recognize that instead of a central controller SHRUTI uses "distributed control mechanisms" as part of its representational machinery. As explained in section 9.1, these mechanisms obviate the need for a central controller that directs the activity of every node at each time step.

NOTES

1. SHRUTI is not an acronym but a Sanskrit work which refers to the oral tradition of communicating knowledge.

2. Bienenstock (personal communication) had also advocated the use of aperiodic synchronous activity over periodic activity.

3. Lado et al. (1992) report finding synchronous oscillations in the motor and sensory cortices during the execution of simple hand movements by analyzing magneto-encephalography data.

4. As argued in Thorpe and Imbert (1989), there are about 100,000 distinct objects that can be named rapidly by people. Hence the number of potential argument fillers is going to be at least 100,000 if not more.

5. Note, however, that if one is only seeking local rather than global consistency then it is possible to seek local support and find a locally consistent hypothesis in a *reflexive* manner. An example of this may be found in the abductive reasoning system described by Ajjanagadde (1991).

6. There are notable exceptions, such as the approach expounded by Newell (1990; see also multiple book review, *BBS* 15(3) 1992). **Feldman,** although not a cognitive psychologist, has long emphasized that the biological architecture places strong computational constraints on the nature of cognitive models. A good example of this is his well-known "hundred step" argument (Feldman & Ballard 1982). [See also Feldman's "Four Frames Suffice" *BBS* 8(2) 1985.]

7. In the case of a query, the possible inferences correspond to the possible derivations of the query. As in the forward case, only derivations that do not violate WMRR capacity and depth bounds are possible.

8. These types of rules have been proposed for linking syntactic structures with conceptual structures within a SHRUTI-like framework (Shastri 1992).

9. It is possible to incorporate certain types of rules into our behavior quite rapidly. Consider "hit the left button if you see an X on the screen." Such rules, however, seem to involve a fairly direct mapping between perception and action.

10. This would be a much longer time than the time a fact may stay active in the WMRR via temporal synchrony but much shorter than the time a fact may stay in medium-term memory.

11. The use of distributed representations and coarse coding by a model does not imply that the model is cognitively significant.

References

Letters *a* and *r* appearing before authors' initials refer to target article and response respectively.

Aaronson, J. (1991) Dynamic fact communication mechanism. A connectionist interface. *Proceedings of the Thirteenth Conference of the Cognitive Science Society.* Erlbaum. [aLS]

Abeles, M. (1982) *Local cortical circuits: Studies of brain function*, vol 6. Springer. [arLS]

(1991) *Corticonics: Neural circuits of the cerebral cortex.* Cambridge University Press. [aLS, WJF]

Ajjanagadde, V. G. (1990) Reasoning with function symbols in a connectionist system. *Proceedings of the Twelfth Conference of the Cognitive Science Society.* Erlbaum [aLS]

(1991) Abductive reasoning in connectionist networks: Incorporating variables, background knowledge, and structured explananda. *Technical Report WSI-91-7.* Wilhelm-Schickard-Institute, University of Tubingen, Germany [arLS, GH, DST]

Ajjanagadde, V. G. & Shastri, L. (1989) Efficient inference with multiplace predicates and variables in a connectionist system. *Proceedings of the Eleventh Conference of the Cognitive Science Society.* Erlbaum. [aLS]

Allen, J. F. (1987) *Natural language understanding.* Benjamin Cummings. [aLS]

Allen, J. F. & Perrault, C. R. (1980) Analyzing intention in utterances. *Artificial Intelligence* 15:143–78. [GH]

Anderson, J. R. (1983) *The architecture of cognition.* Harvard University Press. [aLS]

Baddeley, A. (1986) *Working memory.* Clarendon Press. [arLS, SS]

Bair, W., Koch, C., Newsome, W., Britten, K. & Niebur, E. (1992) Power spectrum analysis of MT neurons from awake monkey. *Society for Neuroscience Abstracts* 18(1):11.12. [MPY]

Barnden, J. A. (1992) Connectionism, generalization and propositional attitudes: A catalogue of challenging issues. In: *The symbolic and connectionist paradigms: Closing the gap,* ed. J. Dinsmore. Erlbaum. [JAB]

Barnden, J. A. & Srinivas, K. (1991) Encoding techniques for complex information structures in connectionist systems. *Connection Science* 3(3):263–309. [aLS, JAB]

Barnes, D. & Hampson, P. J. (1992) Stimulus equivalence, relational frame theory and connectionism: Implications for behaviour analysis and cognitive science. *Proceedings of the Fifteenth Symposium on Quantitative Analyses of Behavior.* Harvard University Press. [PJH]

Bartlett, F. C. (1934) *Remembering* (2nd. ed. 1967). Cambridge University Press. [WJF]

Bibel, W. (1988) Advanced topics in automated deduction. In: *Fundamentals of artificial intelligence II,* ed. R. Nossum. Springer. [SH]

Bienenstock, E. (1991) Notes on the growth of a "composition machine." Presented at the Interdisciplinary Workshop on Compositionality in Cognition and Neural Networks, Abbaye de Royaumont, May. [aLS]

Bobrow, D. & Collins, A., eds. (1975) *Representation and understanding.* Academic Press. [aLS]

Bradski, G., Carpenter, G. A. & Grossberg, S. (1992a) Working memory networks for learning temporal order with application to 3-D visual object recognition. *Neural Computation* 4:270–86. [SG]

(1992b) Working memories for storage and recall of arbitrary temporal sequences. *Proceedings of the International Joint Conferences on Neural Networks,* Piscataway, NJ. [SG]

Braine, M. D. S. (1978) On the relationship between the natural logic of reasoning and standard logic. *Psychological Review* 85:1–21. [MO]

Broadbent, D. E. (1958) *Perception and communication.* Pergamon. [MPY]

Buchanan, B. G. & Shortliffe, E. F. (1984) *Rule-based expert systems: The MYCIN experiments of the Stanford Heuristic Programming Project.* Addison-Wesley. [aLS]

Bylander, T., Allemang, D., Tanner, M. C. & Josephson, J. R. (1991) The computational complexity of abduction. *Artificial Intelligence* 47(1–3):25–60. [aLS]

Cahill, A. & Mitchell, D. C. (1987) Plans and goals in story comprehension. In: *Communication failure in dialogue and discourse,* ed. R. Reilly. Elsevier. [GH]

Carpenter, G. A. & Grossberg, S., eds. (1991) *Pattern recognition by self-organizing neural networks.* MIT Press. [SG]

(1992) A self-organizing neural network for supervised learning, recognition, and prediction. *IEEE Communications* 30:38–49. [SG]

Carpenter, G. A., Grossberg, S., Markuzon, N., Reynolds, J. H. & Rosen, D. B. (1992) Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks* 3:698–713. [SG]

Carpenter, G. A., Grossberg, S. & Reynolds, J. H. (1991) ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network. *Neural Networks* 4:565–88. [SG]

Carpenter, P. A. & Just, M. A. (1977) Reading comprehension as eyes see it. In: *Cognitive processes in comprehension,* ed. M. A. Just & P. A. Carpenter. Erlbaum. [aLS]

Celebrini, S., Thorpe, S., Trotter, Y. & Imbert, M. (1993) Dynamics of orientation coding in area V1 of the awake primate. *Visual Neuroscience* (in press). [SJT]

Chalmers, D. J. (1990) Syntactic transformations on distributed representations. *Connection Science* 1&2(2):53–62. [GD, JWG]

Charniak, E. (1976) *Inference and knowledge* (I and II). In: *Computational semantics,* ed. E. Charniak & Y. Wilks. North-Holland. [aLS]

(1983) Passing markers: A theory of contextual influence in language comprehension. *Cognitive Science* 7:171–90. [aLS, GH]

Chater, N. & Oaksford, M. (1993) *Logicism, mental models and everyday* reasoning: Reply to Garnham. *Mind & Language* 8 (in press). [MO]

Churchland, P. S., Koch, C. & Sejnowski, T. J. (1989) What is computational neuroscience? In: *Computational neuroscience,* ed. E. Schwartz. MIT Press. [MO]

Closman, G. (1988) A model of categorization and learning in a connectionist broadcast system. Ph.D. dissertation, Department of Computer Science, Indiana University. [aLS]

Cohen, P. R., Morgan, J. & Pollack, M. E., eds. (1990) *Intentions in communication.* MIT Press. [GH]

Collins, A. & Michalski, R. (1989) The logic of plausible reasoning: A core theory. *Cognitive Science* 13(1):1–50. [rLS, DST]

Cooke, N. J. (1992) Modeling human expertise in expert systems. In: *The psychology of expertise: Cognitive research and empirical artificial intelligence,* ed. R. R. Hoffman. Springer. [GH]

Cooper, P. R. (1992) Structure recognition by connectionist relaxation: Formal analysis. *Computational Intelligence* 8(1):25–44. [PRC]

Cooper, P. R. & Swain, M. J. (1992) Arc consistency: Parallelism and domain dependence. *Artificial Intelligence* 58:207–35. [PRC]

Corriveau, J. (1991) Time-constrained memory for reader-based text comprehension. *Technical Report CSRI-246.* Ph.D. dissertation, Computer Science Research Institute, University of Toronto. [aLS]

Cottrell, G. (1985) Parallelism in inheritance hierarchies with exceptions. *Proceedings of the Eighth International Joint Conference on Artificial Intelligence,* Los Angeles, CA. [GWC]

(1989) *A connectionist approach to word sense disambiguation.* Pitman. [GWC]

Creutzfeldt, O., Ojemann, G. & Lettich, E. (1989) Neuronal activity in the human lateral temporal lobe. 1. Responses to speech. *Experimental Brain Research* 77:451–75. [SJT]

Crick, F. (1984) Function of the thalamic reticular complex: The searchlight hypothesis. *Proceedings of the National Academy of Sciences* 81:4586–90. [aLS]

Crick, F. & Koch, C. (1990a) Towards a neurobiological theory of consciousness. *Seminars in Neurosciences* 2:263–75. [aLS]

(1990b) Some reflections on visual awareness. *Cold Spring Harbor Symposium on Quantitative Biology* 55:953–62. [SG]

Damasio, A. R. (1989) Time-locked multiregional retroactivation: A systems-level proposal for the neural substrates of recall and recognition. *Cognition* 33:25–62. [aLS]

Davis, P. (1990) Application of optical chaos to temporal pattern search in a nonlinear optical resonator. *Japanese Journal of Applied Physics* 29:L1238–40. [IT]

Dawson, M. R. W. & Schopflocher, D. P. (1992) Autonomous processing in parallel distributed processing networks. *Philosophical Psychology* 5:199–219. [MRWD]

Dehaene, S. & Changeux, J-P. (1991) The Wisconsin card sorting test: Theoretical analysis and modeling in a neuronal network. *Cerebral Cortex* 1:62–79. [MO]

Diederich, J. (1992) Inkrementelles Konnektionistisches Lernen. Forthcoming habilitation thesis, Department of Computer Science, University of Hamburg. [JD]

Dietz, P., Krizanc, D., Rajasekaran, S. & Shastri, L. (1993) A lower bound result for the common element problem and its implication for reflexive reasoning. Technical Report, Department of Computer and Information Science, University of Pennsylvania (forthcoming). [rLS]

Dolan, C. P. & Smolensky, P. (1989) Tensor product production system: A modular architecture and representation. *Connection Science* 1:53–68. [aLS, GSH, RR]

Dorffner, G. & Rotter, M. (1992) On the virtues of functional connectionist compositionality. *Proceedings of the Tenth European Conference on Artificial Intelligence,* ed. B. Neumann. Wiley. [GD]

Dosher, B. A. & Corbett, A. T. (1982) Instrument inferences and verb schemata. *Memory and Cognition* 10(6):531–39. [GH]

Douglas, R. J., Martin, K. A. C. & Whitteridge, D. (1991) An intracellular analysis of the visual responses of neurons in cat visual cortex. *Journal of Physiology* 440:659–96. [RE]

Downing, P. (1977) On the creation and use of English compound nouns. *Language* 53(4):810–42. [GH]

Dwork, C., Kannelakis, P. C. & Mitchell, J. C. (1984) On the sequential nature of unification. *Journal of Logic Programming* 1:35–50. [SH]

Dyer, M. (1983) *In-depth understanding: A computer model of integrated processing for narrative comprehension.* MIT Press. [aLS]

Eckhorn, R. (1991) Stimulus-specific synchronizations in the visual cortex: Linking of local features into global figures? In: *Neuronal cooperativity,* ed. J. Kruger. Springer. [SJT]

Eckhorn, R., Bauer, R., Jordan, W., Brosch, M., Kruse, W., Munk, M. & Reitboeck, H. J. (1988) Coherent oscillations: A mechanism of feature linking in the visual cortex? Multiple electrode and correlation analyses in the cat. *Biological Cybernetics* 60:121–30. [aLS, RE, WJF, IT, SJT]

Eckhorn, R., Gruesser, O.-J., Kroeller, J., Pellnitz, K. & Poepel, B. (1976) Efficiency of different neural codes: Information transfer calculations for three different neural systems. *Biological Cybernetics* 22:49–60. [RE]

Eckhorn, R. & Poepel, B. (1975) Rigorous and extended application of information theory to the afferent visual system of the cat. *Biological Cybernetics* 17:7–17. [RE]

Eckhorn, R., Reitboeck, H. J., Arndt, M. & Dicke, P. (1990) Feature linking via synchronization among distributed assemblies: Simulations of results from cat visual cortex. *Neural Computation* 2:293–307. [aLS, RE]

Eichenbaum, H., Wiener, S. I., Shapiro, M. L. & Cohen, N. J. (1989) The organization of spatial coding in the hippocampus: A study of neural ensemble activity. *Journal of Neuroscience* 9:2764–75. [GWS]

Elman, J. (1991) Distributed representations, simple recurrent networks, and grammatical structure. *Machine Learning* 7:195–225. [JWG]

Engel, A. K., Koenig, P., Gray, C. M. & Singer, W. (1990) Stimulus-dependent neuronal oscillations in cat visual cortex: Intercolumnar interactions as determined by cross-correlation analysis. *European Journal of Neuroscience* 2:588–606. [WJF, MPY]

Engel, A. K., Koenig, P., Kreiter, A. K., Gray, C. M. & Singer, W. (1991) Temporal coding by coherent oscillations as a potential solution to the binding problem: Physiological evidence. In: *Nonlinear dynamics and neural networks*, ed. H. G. Schuster & W. Singer. Weinheim. [aLS]

Engel, A. K., Kreiter, A. K. & Singer, W. (1992) Oscillatory responses in the superior temporal sulcus of anesthetized macaque monkeys. *Society for Neuroscience Abstracts* 18:11.10. [rLS]

Etherington, D. & Reiter, R. (1983) On inheritance hierarchies with exceptions. *Proceedings of the National Conference on Artificial Intelligence*, Washington, D.C. [GWC]

Evans, J. St. B. T. (1972) Interpretation and matching bias in a reasoning task. *Quarterly Journal of Experimental Psychology* 24:193–99. [MO]

(1982) *The psychology of deductive reasoning*. Routledge & Kegan Paul. [MO]

(1983) Linguistic determinants of bias in conditional reasoning. *Quarterly Journal of Experimental Psychology* 35A:635–44. [MO]

(1989) *Bias in human reasoning: Causes and consequences*. Erlbaum. [MO]

Fahlman, S. E. (1979) *NETL: A system for representing real-world knowledge*. MIT Press. [aLS, GH, MO]

(1981) Representing implicit knowledge. In: *Parallel models of associative memory*, ed. G. E. Hinton & J. A. Anderson. Erlbaum. [aLS, MO]

Fahlman, S. E., Hinton, G. E. & Sejnowski, T. J. (1983) Massively parallel architectures for AI: NETL, thistle, and Boltzmann machines. *Proceedings of the National Conference on Artificial Intelligence*. Morgan Kaufmann. [DST]

Fahlman, S. E., Touretzky, D. S. & van Roggen, W. (1981) Cancellation in a parallel semantic network. *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*. Morgan Kaufmann. [aLS]

Feldman, J. A. (1982) Dynamic connections in neural networks. *Biological Cybernetics* 46:27–39. [aLS, PRC]

(1985) Four frames suffice: A provisional model of vision and space. *Behavioral and Brain Sciences* 8:265–89. [PRC]

(1989) Neural representation of conceptual knowledge. In: *Neural connections, mental computation*, ed. L. Nadel, L. A. Cooper, P. Culicover & R. M. Harnish. MIT Press. [aLS]

Feldman, J. A. & Ballard D. H. (1982) Connectionist models and their properties. *Cognitive Science* 6(3):205–54. [aLS, PRC]

Fodor, J. A. (1983) *Modularity of mind*. MIT Press. [MO]

Fodor, J. A. & Pylyshyn, Z. W. (1988a) Connectionism and cognitive architecture: A critical analysis. In: *Connections and symbols*, ed. S. Pinker & J. Mehler, MIT Press. [aLS, DLM]

(1988b) Connectionism and cognitive architecture: A critical analysis. *Cognition* 28:3–71. [GD, MO]

Freeman, M. J. (1975) *Mass action in the nervous system*. Academic Press. [WJF]

(1981) A physiological hypothesis of perception. *Perspectives in Biology and Medicine* 24(4):561–92. [aLS]

(1987) Simulation of chaotic EEG patterns with a dynamic model of olfactory system. *Biological Cybernetics* 56:139–50. [IT]

(1991) The physiology of perception. *Scientific American* 264:78–85. [WJF, IT]

Freeman, W. J. & van Dijk, B. (1987) Spatial patterns of visual cortical fast EEG during conditioned reflex in a rhesus monkey. *Brain Research* 422:267–76. [WJF]

Frisch, A. M. & Allen, J. F. (1982) Knowledge retrieval as limited inference. In: *Notes in computer science: Sixth conference on automated deduction*, ed. D. W. Loveland. Springer. [aLS]

Garnham, A. (1993) Is logicist cognitive science possible? *Mind & Language* 8 (in press). [MO]

Gawne, T. J., Eskandar, E. N., Richmond, B. J. & Optican, L. M. (1991) Oscillations in the responses of neurons in inferior temporal cortex are not driven by stationary visual stimuli. *Society for Neuroscience Abstracts* 17(1):180.18. [MPY]

Geib, C. (1990) A connectionist model of medium-term memory. Term report, Department of Computer and Information Science, University of Pennsylvania. [aLS]

Geller, J. & Du, C. (1991) Parallel implementation of a class reasoner. *Journal of Theoretical Artificial Intelligence* 3:109–27. [aLS]

Genesereth, M. R. & Nilsson, N. J. (1987) *Logical foundations of artificial intelligence*. Morgan Kaufmann. [aLS]

Gerstein, G. L. (1970) Functional association of neurons: Detection and interpretation. In: *The neurosciences: Second study program*, ed. F. O. Schmitt. Rockefeller University Press. [aLS]

Gibbs, R. W., Jr. (1983) Do people always process the literal meanings of indirect requests? *Journal of Experimental Psychology: Learning, Memory, and Cognition* 9:524–33. [GH]

Gilbert, C. D. & Wiesel, T. (1992) Receptive field dynamics in adult primary visual cortex. *Nature* 356:150–52. [JD]

Gray, C. M., Engel, A. K., Koenig, P. & Singer, W. (1991) Properties of synchronous oscillatory neuronal interactions in cat striate cortex. In: *Nonlinear dynamics and neural networks*, ed. H. G. Schuster & W. Singer. Weinheim: VCH Publishers. [aLS, RE]

Gray, C. M., Koenig, P., Engel, A. K. & Singer, W. (1989) Oscillatory responses in cat visual cortex exhibit inter-columnar synchronization which reflects global stimulus properties. *Nature* 338:334–37. [aLS, IT, SJT]

Gray, C. & Singer, W. (1989) Stimulus-specific neural oscillations in orientation specific columns of the visual cortex. *Proceedings of the National Academy of Science* 86:1698–1702. [aLS, RE]

Gross, H., Koerner, E., Boehme, H. & Pomierski, T. (1992) A neural network hierarchy for data and knowledge controlled selective visual attention. In: *Artificial neural networks, 2*, ed. I. Aleksander & J. Taylor. North-Holland. [EK]

Grossberg, S. (1976) Adaptive pattern classification and universal recoding, II: Feedback, expectation, olfaction, and illusions. *Biological Cybernetics* 23:187–202. [SG]

(1978) A theory of visual coding, memory, and development. In: *Formal theories of visual perception*, ed. E. Leeuwenberg & H. Buffart. Wiley. [SG]

ed.(1987a) *The adaptive brain*, vols. 1 & 2. Elsevier/North-Holland. [SG]

(1987b) Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science* 11:23–63. [MRWD]

(1988) *Neural networks and natural intelligence*. MIT Press. [SG]

Grossberg, S. & Mingolla, E. (1985a) Neural dynamics of form perception: Boundary completion, illusory figures, and neon color spreading. *Psychological Review* 92:173–211. [SG]

(1985b) Neural dynamics of perceptual grouping: Textures, boundaries, and emergent segmentations. *Perception & Psychophysics* 38:141–71. [SG]

Grossberg, S. & Somers, D. (1991) Synchronized oscillations during cooperative feature linking in a cortical model of visual perception. *Neural Networks* 4:453–66. [SG]

(1992) Synchronized oscillations for binding spatially distributed feature codes into coherent spatial patterns. In: *Neural networks for vision and image processing*, ed. G. A. Carpenter & S. Grossberg. MIT Press. [SG]

Guha, R. V. & Lenat, D. B. (1990) Cyc: A mid-term report. *Artificial Intelligence Magazine* 11(3):32–59. [aLS]

Halford, G. S., Wilson, W. H., Guo, J., Gayler, R. W., Wiles, J. & Stewart, J. E. M. (1993) Connectionist implications for processing capacity limitations in analogies. In: *Advances in connectionist and neural computational theory*, vol. 2, ed. K. J. Holyoak & J. A. Barnden. Ablex. [GSH]

Hanson, S. J. & Kegl, J. (1987) PARSNIP: A connectionist network that learns natural language grammar from exposure to natural language sentences. *Proceedings of the Ninth Annual Cognitive Science Society Conference*, Seattle, Wa. [GWC]

Harnad, S. (1990) The symbol grounding problem. *Physica D* 42:335–46. [rLS]

Hatfield, H. (1991) Representation and rule-instantiation in connectionist networks. In: *Connectionism and the philosophy of mind*, ed. T. Horgan & J. Tienson. Kluwer Academic. [arLS]

Hayes, P. J. (1977) In defense of logic. *Proceedings of the Fifth Annual International Joint Conference on Artificial Intelligence*, Cambridge, MA. [GWC]

Hayes, S. C. (1991) A relational control theory of stimulus equivalence. In: *Rule-governed behavior: Cognition, contingencies and instructional control*, ed. L. J. Hayes & P. N. Chase. Plenum. [PJH]

Hebb, D. O. (1949) *The organization of behavior.* Wiley. [aLS, GWC, GP]

Henderson, J. (1992) A connectionist parser for structure unification grammar. *Proceedings of the Thirtieth Annual Meeting of the Association of Computational Linguistics.* Association of Computational Linguistics. [arLS]

Hendler, J. (1987) *Integrating marker-passing and problem solving: A spreading activation approach to improved choice in planning.* Erlbaum. [aLS, GH, MO]

Hinton, G. E. (1981) Implementing semantic networks in parallel hardware. In: *Parallel models of associative memory,* ed. G. E. Hinton & J. A. Anderson. Erlbaum. [aLS]

(1986) Learning distributed representations of concepts. *Proceedings of the Eighth Annual Conference of the Cognitive Science Society.* Erlbaum. [GD]

Hirst, G. (1987) *Semantic interpretation and the resolution of ambiguity.* Cambridge University Press. [aLS, GH]

(1988) Resolving lexical ambiguity computationally with spreading activation and polaroid words. In: *Lexical ambiguity resolution,* ed. S. Small, G. Cottrell & M. K. Tanenhaus. Morgan Kaufmann. [GH]

Hirst, G. & Charniak, E. (1982) Word sense and case slot disambiguation. *Proceedings of the Second National Conference on Artificial Intelligence,* Pittsburgh. [GH]

Holden, A. V. & Kryukov, V. I., eds. (1991) *Neurocomputers and attention I & II. Proceedings in nonlinear science.* Manchester University Press. [IT]

Hölldobler, S. (1990) CHCL: A connectionist inference system for Horn logic based on the connection method and using limited resources. *Technical Report 90-042,* International Computer Science Institute, Berkeley, CA. [aLS, SH]

Horn, D., Sagi, D. & Usher, M. (1991) Segmentation, binding, and illusory conjunctions. *Neural computation* 3(4):510–25. [aLS]

Hubel, D. H. & Wiesel, T. N. (1962) Receptive fields, binocular interaction and functional architectures of the cat's visual cortex. *Journal of Physiology* 160:106–54. [WJF]

Hummel, J. E. & Biederman, I. (1992) Dynamic binding in a neural network for shape recognition. *Psychological Review* 99:480–517. [aLS, SJT]

Hummel, J. E., Burns, B. & Holyoak, K. J. (in press) Analogical mapping by dynamic binding: Preliminary investigations. In: *Advances in connectionist and neural computation theory,* vol. 2: *Analogical connections,* ed. K. J. Holyoak & J. A. Barnden. Ablex. [GSH, JEH]

Hummel, J. E. & Holyoak, K. J. (1992) Indirect analogical mapping. *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society.* Erlbaum. [JEH]

Humphreys, M. S., Bain, J. D. & Pike, R. (1989) Different ways to cue a coherent memory system: A theory for episodic, semantic and procedural tasks. *Psychological Review* 96(2):208–33. [GSH]

Ikeda, K., Otsuka, K. & Matsumoto, K. (1989) Maxwell-Bloch turbulence. *Progress of Theoretical Physics* (suppl.)99:295–324. [IT]

James, W. (1890) *Psychology (Briefer course).* Holt. [GWC]

Johannesma, P., Aertsen, A., Vanden Boogaard, H., Eggermont, J. & Epping, W. (1986) From synchrony to harmony: Ideas on the function of neural assemblies and on the interpretation of neural synchrony. In: *Brain theory,* ed. G. Palm & A. Aertsen. Springer. [GP]

Johnson-Laird, P. N. (1983) *Mental models.* Cambridge University Press. [MIB, MO]

(1988) *The computer and the mind.* Harvard University Press. [MIB]

Johnson-Laird, P. N. & Byrne, R. M. J. (1991) *Deduction.* Erlbaum. [MO]

Just, M. A. & Carpenter, P. A., eds. (1977) *Cognitive processes in comprehension.* Erlbaum. [aLS]

Kaneko, K. (1989) Pattern dynamics in spatio-temporal chaos. *Physica* 34D:1–41. [IT]

(1990) Clustering, switching, hierarchical ordering and control in a network of chaotic elements. *Physica* 41D:137–72. [IT]

Kautz, H. A. & Selman, B. (1991) Hard problems for simple default logics. *Artificial Intelligence* 47(1–3):243–79. [aLS]

Keenan, J. M., Baillet, S. D. & Brown, P. (1984) The effects of causal cohesion on comprehension and memory. *Journal of Verbal Learning and Verbal Behavior* 23:115–26. [aLS]

Kintsch, W., ed. (1974) *The representation of meaning in memory.* Erlbaum. [aLS]

(1988) The role of knowledge discourse comprehension: A construction-integration model. *Psychological Review* 95:163–82. [aLS]

Kiper, D. C., Gegenfurtner, K. R. & Movshon, J. A. (1991) The effect of 40Hz flicker on the perception of global stimulus properties. *Society for Neuroscience Abstracts* 17(2):479.4 [MPY]

Klayman, J. & Ha, Y. (1987) Confirmation, disconfirmation, and information in hypothesis testing. *Psychological Review* 94:211–28. [SS]

Koerner, E. & Boehme, H. (1991) Organization of an episodic knowledge base in a neural network architecture with parallel-sequential processing modes for autonomous recognition and learning. In: *Artificial neural networks,* ed. T. Kohonen, K. Mackisara, O. Simula & J. Kangas. Elsevier/North-Holland. [EK]

Koerner, E., Gross, H. & Boehme, H. (1991) Elementary cognitive mechanisms for knowledge based image interpretation. In: *Proceedings of the International Workshop on Adaptive Learning and Neural Networks,* ed. P. Bock, M. Loew, F. J. Radermacher & M. M. Richter. Ulm: Forschungsinstitut für Anwendungs orientierte Wissensrerarbeitung. [EK]

Koerner, E., Gross, H. & Tsuda, I. (1990) Holonic processing in a model system of cortical processors. In: *Biological complexity and information,* ed. H. Shimizu. World Scientific. [EK]

Koerner, E., Salevski, H., Shimizu, H., Koerner, U. & Seifert, S. (submitted) *A structured neural network model of hippocampus and its function as a nonspecific controller of cortical decision making and nontrivial learning.* [EK]

Koerner, E., Shimizu, H. & Tsuda, I. (1987) Parallel in sequence: Towards the architecture of an elementary cortical processor. In: *Parallel algorithms and architectures,* ed. A. Albrecht, H. Hung & G. Mehlhorn. Akademie-Verlag. [EK]

Koerner, E., Tsuda, I. & Shimizu, H. (1987) Take-grant control, variable byte formation and processing parallel in sequence: Characteristics of a new type of holonic processor. In: *Parallel algorithms and architecture,* ed. A. Albrecht, H. Jung & G. Mehlhorn. Springer. [IT]

Kosslyn, S. M., Murphy, G. L., Bemesderfer, M. E. & Feinstein, K. J. (1977) Category and continuum in mental comparisons. *Journal of Experimental Psychology: General* 106:341–75. [PJH]

Kreiter, A. K., Engel, A. K. & Singer, W. (1992) Stimulus dependent synchronization in the caudal superior temporal sulcus of macaque monkeys. *Society for Neuroscience Abstracts* 18:11.11. [rLS]

Kreiter, A. K. & Singer W. (1992) Oscillatory neuronal responses in the visual cortex of the awake macaque monkey. *European Journal of Neuroscience* 4:369–75. [aLS, IT]

Kruse, W., Eckhorn, R., Schanze, T. & Reitboeck, H. J. (1992) Stimulus-induced oscillatory synchronization is inhibited by stimulus-locked non-oscillatory synchronization in cat visual cortex: Two modes that might support feature linking. *Society for Neuroscience Abstracts* 18:131.3. [RE]

Kuramoto, Y. (1991) Collective synchronization of pulse-coupled oscillators and excitable units. *Physica* 50D:15–30. [IT]

Lado, F., Ribary, U., Ioannides, A., Volkman, J., Joliot, M., Mogilner, A. & Llinás, R. (1992) Coherent oscillations in motor and sensory cortices detected using MEG and MFT. *Society for Neuroscience Abstracts* 18:355.15. [rLS]

Lakoff, G. (1987) *Women, fire, and dangerous things: What categories reveal about the mind.* University of Chicago Press. [aLS, SS, DST]

Lakoff, G. & Johnson, M. (1980) *Metaphors we live by.* University of Chicago Press. [DST]

Lange, T. E. & Dyer, M. G. (1989) High-level inferencing in a connectionist network. *Connection Science* 1(2):181–217. [aLS, JAB, GWC]

Lehnert, W. G. & Ringle, M. H., eds. (1982) *Strategies for natural language processing.* Erlbaum. [aLS]

Lettvin, J. Y., Maturana, H. R., McCulloch, W. S. & Pitts, W. H. (1959) What the frog's eye tells the frog's brain. *Proceedings of the Institute of Radio Engineering* 47:1940–51. [WJF]

Levesque, H. J. (1988) Logic and the complexity of reasoning. *Journal of Philosophical Logic* 17:335–89. [aLS]

Levesque, H. J. & Brachman, R. J. (1985) A fundamental tradeoff in knowledge representation and reasoning. In: *Readings in knowledge representation,* ed. R. J. Brachman & H. J. Levesque. Morgan Kaufmann. [aLS, GWC]

Levi, J. N. (1978) *The syntax and semantics of complex nominals.* Academic Press. [GH]

Livingstone, M. S. (1991) Visually evoked oscillations in monkey striate cortex. *Society for Neuroscience Abstracts* 17:73.3. [rLS]

Lucas, M. M., Tanenhaus, M. K. & Carlson, G. N. (1990) Levels of representation in the interpretation of anaphoric reference and instrument inference. *Memory and Cognition* 18(6):611–31. [GH]

Lynch, G. (1986) *Synapses, circuits, and the beginnings of memory.* MIT Press. [aLS]

MacVicar, B. & Dudek, F. E. (1980) Dye-coupling between CA3 pyramidal cells in slices of rat hippocampus. *Brain Research* 196:494–97. [aLS]

Malloch, M. I., Oaksford, M. & Iddon, J. (1992) Impairments of reasoning, memory and planning in early stage Parkinsonism. *Technical Report No. UWBCNU-TR-13,* Cognitive Neurocomputation Unit, University of Wales, Bangor. [MO]

Mandelbaum, R. (1991) A robust model for temporal synchronization of distant nodes: Description and simulation. Term Report, Department of Computer and Information Science, University of Pennsylvania. [aLS]

Mandelbaum, R. & Shastri, L. (1990) A robust model for temporal synchronisation of distant nodes. (Unpublished report.) [aLS]

Mani, D. R. & Shastri, L. (1991) Combining a connectionist type hierarchy with a connectionist rule-based reasoner. *Proceedings of the Thirteenth Conference of the Cognitive Science Society*. Erlbaum. [aLS]

(1992) A connectionist solution to the multiple instantiation problem using temporal synchrony. *Proceedings of the Fourteenth Conference of the Cognitive Science Society*. Erlbaum. [aLS]

Marr, D. (1971) Simple memory: A theory for archicortex. *Philosophical Transactions of the Royal Society* B 262:23–81. [aLS]

Martin, C. E. & Riesbeck, C. K. (1986) Uniform parsing and inferencing for learning. *Proceedings of the Fifth National Conference on Artificial Intelligence*, Philadelphia. [GH]

Matsumoto, K. & Tsuda, I. (1987) Extended information in one-dimensional maps. *Physica* 26D:347–57. [IT]

McAllester, D. A. (1990) *Automatic recognition of tractability in inference relations. Memo 1215*, MIT Artificial Intelligence Laboratory. [aLS]

McCarthy, J. (1988) Epistemological challenges for connectionism [Commentary on Smolensky]. *Behavioral and Brain Sciences* 11(1):44. [aLS]

McDermott, D. (1981) Artificial intelligence meets natural stupidity. In: *Mind design*, ed. J. Haugland. MIT Press/Bradford Books. [DLM]

(1986) A critique of pure reason. Technical Report, Department of Computer Science, Yale University. [MO]

McKendall, T. (1991) A design for an answer extraction and display scheme for a connectionist rule-based reasoner. Unpublished report on work done for National Science Foundation, Research Experience for Undergraduates grant IRI 88-05465. [aLS]

McKoon, G. & Ratcliff, R. (1980) The comprehension processes and memory structures involved in anaphoric reference. *Journal of Verbal Learning and Verbal Behavior* 19:668–82. [aLS]

(1981) The comprehension processes and memory structures involved in instrumental inference. *Journal of Verbal Learning and Verbal Behavior* 20:671–82. [aLS]

(1986) Inferences about predictable events. *Journal of Experimental Psychology: Learning, Memory, and Cognition* 12:82–91. [aLS]

McMillan, C., Mozer, M. & Smolensky, P. (1991): The connectionist scientist game. *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*. Erlbaum. [GD]

McRoy, S. W. (1993) Abductive interpretation and re-interpretation of natural language utterances. Ph.D. dissertation, Department of Computer Science, University of Toronto. [GH]

McRoy, S. W. & Hirst, G. (1993) Abductive explanations of dialogue misunderstanding. *Proceedings, Sixth Conference of the European Chapter of the Association for Computational Linguistics*, Utrecht. [GH]

Merzenich, M. M., Recanzone, G., Jenkins, W. M., Allard, T. T. & Nudo, R. J. (1988) Cortical representational plasticity. In: *Neurobiology of neocortex*, ed. P. Rakie & W. Singer. Wiley. [JD]

Miller, G. A. (1956) The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review* 63(2):81–97 [aLS, GSH, EK]

Milner, B. (1963) Effects of different brain lesions on card sorting. *Archives of Neurology* 9:90–100. [MO]

Minsky, M. (1975) A framework for representing knowledge. In: *The psychology of computer vision*, ed. P. M. Winston. McGraw-Hill. [aLS]

(1985) *The society of mind*. Simon & Schuster. [EK]

Mountcastle, V. B. (1957) Modality and topographic properties of single neurons of cat's somatic cortex. *Journal of Neurophysiology* 20:408–34. [WJF]

Mozer, M. C., Zemel, R. S. & Behrman, M. (1991) Learning to segment images using dynamic feature binding. *Technical Report CU-CS-540-91*, University of Colorado at Boulder. [aLS]

Newell, A. (1980) Harpy, production systems and human cognition. In: *Perception and production of fluent speech*, ed. R. Cole. Erlbaum. [aLS]

(1990) *Unified theories of cognition*. Harvard University Press. [arLS]

Newell, A. & Simon, H. A. (1972) *Human problem solving*. Prentice-Hall. [aLS]

Norman, D. A. & Shallice, T. (1985) Attention to action: Willed and automatic control of behaviour. In: *Consciousness and self-regulation*, vol. 4, ed. R. J. Davidson, G. E. Schwartz & D. Shapiro. Plenum. [MO]

Norvig, P. (1989) Marker passing as a weak method for text inferencing. *Cognitive Science* 113:569–620. [aLS, GH]

Oaksford, M. (1993) Mental models and the tractability of everyday reasoning. *Behavioral and Brain Sciences* 16(2):360–61. [MO]

Oaksford, M. & Chater, N. (1991) Against logicist cognitive science. *Mind & Language* 6:1–38. [MO]

(1992a) Reasoning theories and bounded rationality. In: *Rationality*, ed. K. Manktelow & D. Over. Routledge. [MO]

(1992b) Bounded rationality in taking risks and drawing inferences. *Theory & Psychology* 2:225–30. [MO]

(in press) *Cognition and inquiry*. Academic Press. [MO]

Oaksford, M., Malloch, M. I. & Swain, S. (1992a) Transitive inference in closed head injury: A single case study. *Technical Report No. UWBCNU-TR-12*, Cognitive Neurocomputation Unit, University of Wales, Bangor. [MO]

Oaksford, M., Malloch, M. I., Watson, F. & Hargreaves, I. (1992b) Impairments of reasoning, memory and attention in frontal lobe damage: A single case study. *Technical Report No. UWBCNU-TR-11*, Cognitive Neurocomputation Unit, University of Wales, Bangor. [MO]

Oaksford, M. & Stenning, K. (1992) Reasoning with conditionals containing negated constituents. *Journal of Experimental Psychology: Learning, Memory & Cognition* 18:835–54. [MO]

Oram, M. W. & Perrett, D. I. (1992) Time course of neural responses discriminating different views of the face and head. *Journal of Neurophysiology* 69:70–84. [SJT]

Pabst, M., Reitboeck, H. J. & Eckhorn, R. (1989) A model of preattentive region definition based on texture analysis. In: *Models of brain function*, ed. R. M. J. Cotterill. Cambridge University Press. [RE]

Palm, G. (1982) *Neural assemblies: An alternative approach to artificial intelligence*. Springer. [GP]

(1986) Associative networks and cell assemblies. In: *Brain theory*, ed. G. Palm & A. Aertsen. Springer. [GP]

(1990) *Cell assemblies as a guideline for brain research. Concepts in Neuroscience* 1:133–14. [GP]

Pelletier, F. J. (1982) Completely non-causal, completely heuristically driven, automated theorem proving. *Technical Report* 82-7, Department of Computing Science, University of Alberta. [MRWD]

Pfeifer, R. & Verschure, P. (1992) Beyond rationalism: Symbols, patterns and behavior. *Connection Science* 4:313–25. [JD]

Pollack, J. B. (1988) Recursive auto-associative memory: Devising compositional distributed representations. *Technical report MCCS-88-124*. Computing Research Laboratory, New Mexico State University. [GH]

(1990) Recursive distributed representations. *Artificial Intelligence* 46:77–105. [GD, GH]

Posner, M. I. & Snyder, C. R. R. (1975) Attention and cognitive control. In: *Information processing and cognition: The Loyala Symposium*, ed. R. L. Solso. Erlbaum. [aLS]

Potts, G. R., Keenan, J. M. & Golding, J. M. (1988) Assessing the occurrence of elaborative inferences: Lexical decision versus naming. *Journal of Memory and Language* 27:399–415. [aLS]

Quillian, M. R. (1968) Semantic memory. In: *Semantic information processing*, ed. M. Minsky. MIT Press. [aLS]

Ramesh, R., Verma, R. M., Krishnaprasad, T. & Ramakrishnan, I. V. (1989) Term matching on parallel computers. *Journal of Logic Programming* 6:213–28. [SH]

Reder, L. M. & Ross, B. H. (1983) Integrated knowledge in different tasks: The role of retrieval strategy on fan effects. *Journal of Experimental Psychology: Learning, Memory, and Cognition* 9:55–72. [aLS]

Reitboeck, H. J., Eckhorn, R., Arndt, M. & Dicke, P. (1990) A model for feature linking via correlated neural activity. In: *Synergetics of cognition*. Springer series in synergetics, vol. 45, ed. H. Haken & M. Stadler. Springer. [IT]

Reiter, R. (1980) A logic for default reasoning. *Artificial Intelligence* 13:81–132. [rLS]

Riesbeck, C. R. & Schank, R. C. (1989) *Inside case-based reasoning*. Erlbaum. [PRC]

Rips, L. J. (1983) Cognitive processes in propositional reasoning. *Psychological Review* 90:38–71. [MO]

Rohwer, R. A. (1993) A representation of representation applied to a discussion of variable binding. In: *Neurodynamics and psychology*, ed. M. Oaksford & G. Brown. Academic Press. [RR]

Rolls, E. T. (1991) Neural organisation of higher visual functions. *Current Opinion in Neurobiology* 1:274–78. [aLS, MPY]

Rotter, M. & Dorffner, G. (1990) Struktur und Konzeptrelationen in verteilten Netzwerken. In: *Konnektionismus in Artificial Intelligence und Kognitionsforschung*, ed. G. Dorffner. Springer. [GD]

Rumelhart, D. E. (1989) Toward a microstructural account of human reasoning. In: *Similarity and analogical reasoning*, ed. S. Vosniadou & A. Ortony. Cambridge Universtiy Press. [MO]

Rumelhart, D. E. & McClellan, J. L., eds. (1986) *Parallel distributed processing: Explorations in the microstructure of cognition*, vol 1. Bradford Books/MIT Press. [aLS]

Rumelhart, D. E., Smolensky, P., McClelland, J. L. & Hinton, G. E. (1986) Schemata and sequential thought processes in PDP models. In: *Parallel distributed processing: Explorations in the microstructure of cognition, vol. 2: Psychological and biological processes*, ed. J. L. McClelland & D. E. Rumelhart. MIT Press. [MO]

Schank, R. C. & Abelson, R. P. (1977) *Scripts, plans, goals and understanding*. Erlbaum. [aLS]

Schanze, T. & Eckhorn, R. (1991) Synchronization statistics of stimulus-specific oscillatory events in cat visual cortex. In: *Synapse, transmission, modulation*, ed. N. Elsner & H. Penzlin. Thieme Verlag. [RE]

Schneider, W. & Shiffrin, R. M. (1977) Controlled and automatic human information processing. I. Detection, search, and attention. *Psychological Review* 84:1–66. [aLS]

Schubert, L. K. (1989) An episodic knowledge representation for narrative texts. *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*. Morgan Kaufmann. [aLS]

Sejnowski, T. J. (1981) Skeleton filters in the brain. In: *Parallel models of associative memory*, ed. G. E. Hinton & J. A. Anderson. Erlbaum. [aLS]

Servan-Schreiber, D., Cleeremans, A. & McClelland, J. (1989) Encoding semantical structure in simple recurrent nets. In: *Advances in neural information processing systems 1*, ed. D. Touretzsky. Morgan Kaufmann. [JWG]

Shallice, T. (1982) Specific impairments of planning. *Philosophical Transactions of the Royal Society of London* B 298:199–209. [MO]

Sharkey, N. E. (1992) The causal role of the constituents of superpositional representations. In: *Cybernetics and systems '92*, ed. R. Trappl. World Scientific. [GD]

Shastri, L. (1988a) *Semantic networks: An evidential formulation and its connectionist realization*. Pitman/Morgan Kaufmann. [arLS, PRC]

(1988b) A connectionist approach to knowledge representation and limited inference. *Cognitive Science* 12(3):331–92. [arLS, GWC]

(1990) Connectionism and the computational effectiveness of reasoning. *Theoretical Linguistics* 16(1):65–87. [aLS]

(1991) Relevance of connectionism to AI: A representation and reasoning perspective. In: *Advances in connectionist and neural computation theory*, vol. 1, ed. J. Barnden & J. Pollack. Ablex. [aLS]

(1992) Encoding higher-order bindings in LCS structures. Working notes for the NLQ-Project. National Science Foundation. [rLS]

(1993a) A realization of preference rules using temporal synchrony (in preparation). [aLS]

Shastri, L. (1993b) Learning evidential rules in SHRUTI (in preparation). [rLS]

Shastri, L. & Ajjanagadde, V. G. (1990) A connectionist representation of rules, variables and dynamic binding. *Technical Report MS-CIS-90-05*, Department of Computer and Information Science, University of Pennsylvania. [aLS]

Shastri, L. & Feldman, J. A. (1986) Semantic nets, neural nets, and routines. In: *Advances in cognitive science*, ed. N. Sharkey. Ellis Harwood/Wiley. [arLS]

Shiffrin, R. M. & Schneider, W. (1977) Controlled and automatic human information processing: II. Perceptual learning, automatic attending, and a general theory. *Psychological Review* 84:127–90. [aLS]

Shimizu, H. & Yamaguchi, Y. (1987) Synergetic computers and holonics-information dynamics of semantic computers. *Physica Scripta* 36:970–85. [IT]

Shimizu, H., Yamaguchi, Y., Tsuda, I. & Yano, M. (1985) Pattern recognition based on holonic information dynamics. In: *Complex systems-operational approaches*, ed. H. Haken. Springer Series in Synergetics, vol. 31. [EK]

Singer, M. & Ferreira, F. (1983) Inferring consequences in story comprehension. *Journal of Verbal Learning and Verbal Behavior* 22:437–48. [aLS]

Singer, W. (1987) Activity-dependent self-organization of synaptic connections as a substrate of learning. In: *The neural and molecular bases of learning*, ed. J.-P. Changeux & M. Konishi. Wiley. [JD]

Skarda, C. A. & Freeman, W. J. (1987) How brains make chaos in order to make sense of the world. *Behavioral and Brain Sciences* 10:161–95. [WJF, IT]

Sloman, S. A. (1993) Feature-based induction. *Cognitive Psychology* 25 (in press). [SS]

Smolensky, P. (1988) On the proper treatment of connectionism. *Behavioral and Brain Sciences* 11:1–74. [GD, EK]

(1990) Tensor product variable binding and the representation of symbolic structure in connectionist systems. *Artificial Intelligence* 46(1–2):159–216. [aLS, GSH]

Squire, L. R. (1987) *Memory and brain*. Oxford University Press. [aLS]

Squire, L. R. & Zola-Morgan, S. (1991) The medial temporal lobe memory system. *Science* 253:1380–86. [aLS]

Stenning, K., Shepard, M. & Levy, J. (1988) On the construction of representations for individuals from descriptions in text. *Language and Cognitive Processes* 3(2):129–64. [aLS]

Stevens, C. F. (1989) How cortical interconnectedness varies with network size. *Neural Computation* 1:473–79. [JD]

Stolcke, A. K. & Wu, D. (1992) Tree matching with recursive distributed representations. AAAI-92 Workshop on Integrating Neural and Symbolic Processes, San Jose, CA. (Also available as *Technical Report 92-025*, International Computer Science Institute, Berkeley.) [GH]

Strehler, B. L. & Lestienne, R. (1986) Evidence on precise time-coded symbols and memory of patterns in monkey cortical neuronal spike trains. *Proceedings of the National Academy of Science* 83:9812–16. [aLS]

Strong, G. W. & Whitehead, B. A. (1989) A solution to the tag-assignment problem for neural nets. *Behavioral and Brain Sciences* 12:381–433. [aLS, GWS]

Sumida, R. A. & Dyer, M. G. (1989) Storing and generalizing multiple instances while maintaining knowledge-level parallelism. *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*. Morgan Kaufmann. [aLS]

Thorpe, S. J., Celebrini, S., Trotter, Y. & Imbert, M. (1991) Dynamics of stereo processing in area V1 of the awake primate. *European Journal of Neuroscience* (Suppl.) 4:83. [SJT]

Thorpe, S. J., Celebrini, S., Trotter, Y., Pouget, A. & Imbert, M. (1989) Dynamic aspects of orientation coding in area V1 of the awake primate. *European Journal of Neuroscience* (Suppl.) 2:322. [SJT]

Thorpe, S. J. & Imbert, M. (1989) Biological constraints on connectionist models. In: *Connectionism in perspective*, ed. R. Pfeiffer, Z. Schreter, F. Fogelman-Souile & L. Steels. Elsevier. [arLS, SJT]

Tomabechi, H. & Kitano, H. (1989) Beyond PDP: The frequency modulation neural network approach. *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*. Morgan Kaufmann. [aLS]

Touretzky, D. S. (1986) *The mathematics of inheritance systems*. Morgan Kaufmann/Pitman. [aLS]

(1990) BoltzCONS: Dynamic symbol structures in a connectionist network. *Artificial Intelligence* 46:1–2, 5–46. [rLS, JAB]

Touretzky, D. S. & Hinton, G. E. (1988) A distributed connectionist production system. *Cognitive Science* 12(3):423–66. [aLS, GWC]

Tovee, M. J. & Rolls, E. T. (1992) Oscillatory activity is not evident in the primate temporal visual cortex with static stimuli. *Neuroreport* 3:369–71. [aLS, SJT, MPY]

Toyama, K., Kimura, M. & Tanaka, T. (1981) Cross correlation analysis of interneuronal connectivity in cat visual cortex. *Journal of Neurophysiology* 46(2):191–201. [aLS]

Treisman, A. & Gelade, G. (1980) A feature integration theory of attention. *Cognitive Psychology* 12:97–136. [aLS]

Tsuda, I. (1991) Chaotic itinerancy as a dynamical basis of hermeneutics in brain and mind. *World Futures* 32:167–84. [WJF, IT]

(1992) Dynamic link of memory-chaotic memory map in nonequilibrium neural networks. *Neural Networks* 5:313–26. [IT]

Tulving, E. (1983) *Elements of episodic memory*. Oxford University Press. [aLS]

Tversky, A. & Kahneman, D. (1983) Extensional versus intuitive reasoning: The conjunction fallacy in probability judgment. *Psychological Review* 90:293–315. [SS]

Ullman, J. D. & van Gelder, A. (1988) Parallel complexity of logical query programs. *Algorithmica* 3:5–42. [aLS]

Valiant, L. G. (1988) Functionality in neural nets. *Proceedings of the National Conference on Artificial Intelligence*, Saint Paul, MN. [GWC]

van Gelder, T. (1990) Compositionality: A connectionist variation on a classical theme. *Cognitive Science* 14:208–12. [GD]

(1991) Classical questions, radical answers: Connectionism and the structure of mental representations. In: *Connectionism and the philosophy of mind*, ed. T. Horgan & J. Tienson. Kluwer. [JWG]

Velmans, M. (1991) Is human information processing conscious? [and Commentary thereon]. *Behavioral and Brain Sciences* 14(4):651–726. [GH]

Vogels, R. & Orban, G. A. (1991) Quantitative study of striate single unit responses in monkeys performing an orientation discrimination task. *Experimental Brain Research* 84:1–11. [SJT]

von der Malsburg, C. (1981) The correlation theory of brain function. *Internal Report 81-2*. Department of Neurobiology, Max-Planck-Institute for Biophysical Chemistry, Göttingen, Germany. [aLS, WJF]

(1986) Am I thinking assemblies? In: *Brain theory*, ed. G. Palm & A. Aertsen. Springer. [aLS, GP]

von der Malsburg, C. & Schneider, W. (1986) A neural cocktail-party processor. *Biological Cybernetics* 54:29–40. [aLS, WJF, EK, IT]

Wason, P. C. (1960) On the failure to eliminate hypotheses in a conceptual task. *Quarterly Journal of Experimental Psychology* 12:129–40. [SS]

(1966) Reasoning. In: *New horizons in psychology*, ed. B. Foss. Penguin. [MO]

Whitney, P. & Williams-Whitney, D. (1990) Toward a contextualist view of elaborative inferences. In: *The psychology of learning and motivation*, vol. 25, ed. A. C. Graesser & G. H. Bower. Academic Press. [GH]

Wickelgren, W. A. (1979) Chunking and consolidation: A theoretical synthesis of semantic networks, configuring in conditioning, S-R versus cognitive learning, normal forgetting, the amnesic syndrome, and the hippocampal arousal system. *Psychological Review* 86(1):44–60. [aLS]

Wilensky, R. (1983) *Planning and understanding: A computational approach to human reasoning*. Addison-Wesley. [aLS]

Wu, D. (1989) A probabilistic approach to marker propagation. *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*. Morgan Kaufmann. [GH]

(1992a) Automatic inference: A probabilisitic basis for natural language interpretation. Ph.D. dissertation (*Technical Report UCB/CSD 92/692*). Division of Computer Science, University of California at Berkeley. [GH]

(1992b) Approximate maximum-entropy integration of syntactic and semantic constraints. AAAI-92 Workshop on Statistically-Based NLP Techniques, San Jose, CA. [GH]

Yao, Y., Freeman, W. J., Burke, B. & Yang, Q. (1991) Pattern recognition by a distributed neural network: An industrial application. *Neural Networks* 4:103–12. [WJF]

Young, M. P., Tanaka, K. & Yamane, S. (1991) On oscillating neuronal responses in monkey visual cortex. *Society for Neuroscience Abstracts* 17(1):73.9. [MPY]

Connection Science, Vol. 5, Nos. 3 & 4, 1993

Reflexive Reasoning with Multiple Instantiation in a Connectionist Reasoning System with a Type Hierarchy

D. R. MANI & LOKENDRA SHASTRI

We describe a hybrid knowledge representation and reasoning system that integrates a rule-based reasoner with a type hierarchy and can accommodate multiple dynamic instantiations of predicates. The system—which is an extension of the reasoner described in Shastri and Ajjanagadde (1990)—maintains and propagates variable bindings using temporally synchronous (i.e. in-phase) firing of appropriate nodes, and can perform a broad class of reasoning with extreme efficiency. The type hierarchy allows the system to encode generic facts such as 'cats prey on birds' and rules such as 'if x *preys on* y *then* y *is scared of* x*' and use them to infer that Tweety the canary is scared of Sylvester the cat. The system can also encode qualified rules such as 'if an* animate *agent collides with a* solid object *then the agent gets hurt'. The ability to accommodate multiple dynamic instantiations of any predicate allows the system to handle a much broader class of inferences, including those involving transitivity and bounded recursion. The proposed system can answer queries in time which is independent of the size of the knowledge base, and is only proportional to the length of the shortest derivation of the query.*

KEYWORDS: Binding problem, connectionism, knowledge representation, multiple instantiation, reflexive reasoning, type hierarchy.

1. Introduction

Connectionist networks, or neural networks, have primarily been used to model 'low-level' cognitive phenomena including visual pattern recognition, speech processing and effector control.[1] For these tasks, connectionist networks offer several advantages: massive parallelism, noise- and fault-tolerance, graceful degradation and trainability. On the other hand, classical artificial intelligence (AI) or symbol-processing systems have primarily focused on 'high-level' processes involving 'reasoning' using rules and manipulating abstract knowledge. Though symbol-processing systems are capable of knowledge representation and rule-based reasoning, they have had the disadvantage of not being scalable—these

D. R. Mani, Department of Computer and Information Science, University of Pennsylvania, 200 South 33rd Street, Philadelphia, PA 19104, USA. L. Shastri, International Computer Science Institute, 1947 Center Street, Suite 600, Berkeley, CA 94704, USA. E-mail: mani@linc.cis.upenn.edu and shastri@icsi.berkeley.edu.

systems become slow and unusable as the size of the knowledge base increases. This is especially true when modeling human cognition. Hybrid architectures are an integration of the notions of neural and symbolic processing systems in an attempt to get the best of both worlds.

We describe a knowledge representation and reasoning system which combines notions from classical AI and connectionism. The motivation for developing such a reasoning system, however, is not to 'simulate' classical symbol-processing using connectionism. Instead, the effort is motivated by the belief that an integration of neural and symbolic systems would result in models that retain the essential representational and inferential powers of classical systems, and at the same time constrain and limit the reasoning system in cognitively plausible ways. Furthermore, the simplicity, efficiency and massive parallelism of connectionist models combined with the symbol-processing capabilities of classical AI systems would lead to an efficient and scalable reasoning system.

A crucial factor in the failure of classical symbol-processing systems at modeling human cognition is their use of general-purpose paradigms. From a reasoning perspective, the use of computationally intractable techniques like general-purpose theorem proving will not lead to efficient, rapid and tractable reasoning systems. To make reasoning tractable, constraints and restrictions will have to be imposed in order to limit the reasoning capability of the system in several ways. An *ad hoc* choice of these constraints will result in a system which is tractable, but probably not very useful. By using connectionism as our underlying paradigm, and by turning to cognitive science, psychology and neuroscience for a realistic set of constraints, we hope to develop a model of tractable reasoning that not only retains the essential representational and inferential powers of classical systems, but also limits the resulting system in cognitively plausible ways.

In developing a connectionist knowledge representation and reasoning system, one of the key issues that needs to be addressed is the dynamic variable binding problem (Feldman, 1982; van der Malsburg, 1986). Shastri and Ajjanagadde (1993a, 1990; Ajjanagadde & Shastri, 1991) have described a solution to the variable binding problem and shown that the solution leads to the design of a connectionist reasoning system that can represent systematic knowledge involving *n*-ary predicates and variables, and perform a broad class of reasoning with extreme efficiency. The system can store both specific situations (facts) and general systematic relationships in the domain (rules). The time taken by the reasoning system to draw an inference is only proportional to the length of the chain of inference, and is *independent* of the number of rules and facts encoded by the system. The reasoning system maintains and propagates variable bindings using temporally synchronous—i.e. in-phase—firing of appropriate nodes. The solution to the variable binding problem allows the system to maintain and propagate a large number of bindings simultaneously as long as the number of distinct entities participating in the bindings during any given episode of reasoning remains bounded. Reasoning in the proposed system is the transient but systematic flow of rhythmic patterns of activation, where each phase in the rhythmic pattern corresponds to a distinct entity involved in the reasoning process, and where variable bindings are represented as the synchronous firing of appropriate argument and entity (filler) nodes. A fact behaves as a temporal pattern matcher that becomes 'active' when it detects that the bindings corresponding to it are present in the system's pattern of activity. Rules are interconnection patterns that propagate and transform rhythmic patterns of activity across relational structures.

The system attempts to model efficient, effortless and spontaneous reasoning over a large body of knowledge. Such reasoning has been described as reflexive reasoning (Shastri, 1990). The system as described in Shastri and Ajjanagadde (1990), however, has some limitations. In this paper, we overcome some of these limitations by extending the basic system so that it can effectively draw a wider class of inferences. In particular, we describe how (i) the rule-based component can be interfaced with a type hierarchy and (ii) the inferences drawn by the reasoning system may involve multiple dynamic instantiations of the same predicate (Barnden & Pollack, 1991; Dyer, 1991). The extended system continues to be scalable and can reason rapidly with large knowledge bases. The work also leads to several predictions which have cognitive and psychological implications, and offer fresh insight into the nature of reflexive reasoning (Sections 5.6 and 5.7).

Several other researchers have proposed connectionist knowledge representation and reasoning systems using a variety of techniques (Section 6). These include the use of dynamic connections (Feldman, 1982), parallel constraint satisfaction (Touretzky & Hinton, 1988), position specific encoding (Barnden & Srinivas, 1991), tensor product representations (Dolan & Smolensky, 1989) and signatures (Lange & Dyer, 1989).

1.1. The Need for a Type Hierarchy

Human agents can reason with types, categories or concepts as effectively as they can reason with instances or individuals. If we know that Sylvester is a cat and Tweety is a canary, using the knowledge that 'cats prey on birds', we can spontaneously infer that 'Sylvester preys on Tweety'. Such inferences may be performed efficiently by organizing concepts into a type hierarchy so that we can quickly traverse the hierarchy to find out facts like 'canaries are birds', 'birds and cats are animals', and so on. Though these inferences can be drawn without the use of a type hierarchy (by encoding the type information directly in the rule base), using a separate type hierarchy substantially improves reasoning efficiency, especially since the inferences drawn in the type hierarchy are used repeatedly in reflexive reasoning (also see Section 5).

Interaction between the type hierarchy and the rule base further facilitates inferences that the reasoning system can draw. Continuing with the Tweety–Sylvester example, if we knew the rule 'if x preys on y then y is scared of x' we can infer, using the type hierarchy, that Tweety is scared of Sylvester. Moreover, the type hierarchy allows rule-like knowledge of the form 'cats prey on birds' to be encoded as the fact preys-on(Cat,Bird), and hence allows this knowledge not only to be used during reasoning but also to be retrieved. Without a type hierarchy, this knowledge would be encoded as the rule $\forall x, y$ cat(x) $\wedge$ bird(y) $\Rightarrow$ preys-on(x,y). Consequently, it would participate in reasoning, but would not be retrievable *per se*. A type hierarchy also allows the use of non-specific instances of types and can therefore represent facts like 'there is *a cat* that loves all birds'.

The encoding of context-sensitive rules—where the firing of the rule is dependent on the type of the role fillers—is facilitated by the use of types. For example, if a ball hits a wall, we would not worry about the ball getting hurt. But we would have no difficulty in inferring that John would be hurt if he ran into a wall. Implicitly, we could be thought of as applying the rule 'if an animate agent collides with a solid object, the animate agent would get hurt'. This notion can be formalized by the use of typed variables, and the rule stated as: ∀x:animate, y:solid-obj collide(x,y) ⇒ hurt(x).

In order that our reasoning system be able to reason effectively in the situations mentioned above, we need to augment the rule base with a type hierarchy. A brief outline of how this could be done was described in Shastri and Ajjanagadde (1990). In this paper, we describe a detailed solution to the problem.

1.2. The Need for Multiple Dynamic Instantiations of Predicates

The need for multiple instantiation arises in two situations. The simpler of the two cases—multiple dynamic instantiations of concepts in the type hierarchy—arises when two objects of the same type are being represented. If the system has simultaneously to represent 'cats are animals' and 'birds are animals' in its state of activation, the concept representing animal will have to fire in synchrony with both the cat concept and the bird concept. The more general problem of representing multiple dynamic instantiations of predicates arises during reflexive reasoning as brought out by the following examples. If we know that Mary is John's spouse, we would not have any difficulty in realizing that John is Mary's spouse. In other words, given spouse-of(Mary,John) we can reflexively answer 'yes' to the query spouse-of(John,Mary)? Such behavior would require the spouse-of predicate to be instantiated twice: once with spouse-of(John,Mary) and again with spouse-of(Mary,John). As another example, consider the situation in which we know that Mary is older than John's father. If we now hear that John married Mary, we can instantly sense the unusualness of the situation, since Mary is obviously much older than John. But the fact that Mary is older than John has not been explicitly stated. This would suggest that we may have inferred older-than(Mary,John) using the facts older-than(Mary,John's-father) and older-than(John's-father,John),[2] and the transitive nature of the older-than predicate. To model this scenario in the reasoning system, we would need simultaneously to represent three instantiations of older-than. Similarly, we can, without conscious deliberation, infer that John may be jealous of Tom if we know that John loves Mary and Mary loves Tom. Here again, we would need the ability to represent multiple instantiations of the loves predicate to capture the situation. Thus, a system for modeling reflexive reasoning should be capable of representing multiple instantiations of predicates.

The system described in Shastri and Ajjanagadde (1990) has the limitation that any predicate in the reasoner can be instantiated at most once.[3] In this paper, we describe how this system can be extended to deal with multiple instantiations of predicates in the reasoner as well as multiple instantiations of concepts in the type hierarchy.

1.3. Overview

We begin with a brief overview of the basic rule-based reasoning system (Section 2). The realization of the type hierarchy is described in Section 3, followed by the specification of the multiple instantiation mechanisms in Section 4. Section 5 describes how rules, facts and queries are handled by the extended reasoning system. This section also describes the constraints introduced by the reasoning system and their significance. We then conclude with a brief discussion of related work, the relevance of this work and possible future research directions.

(a)

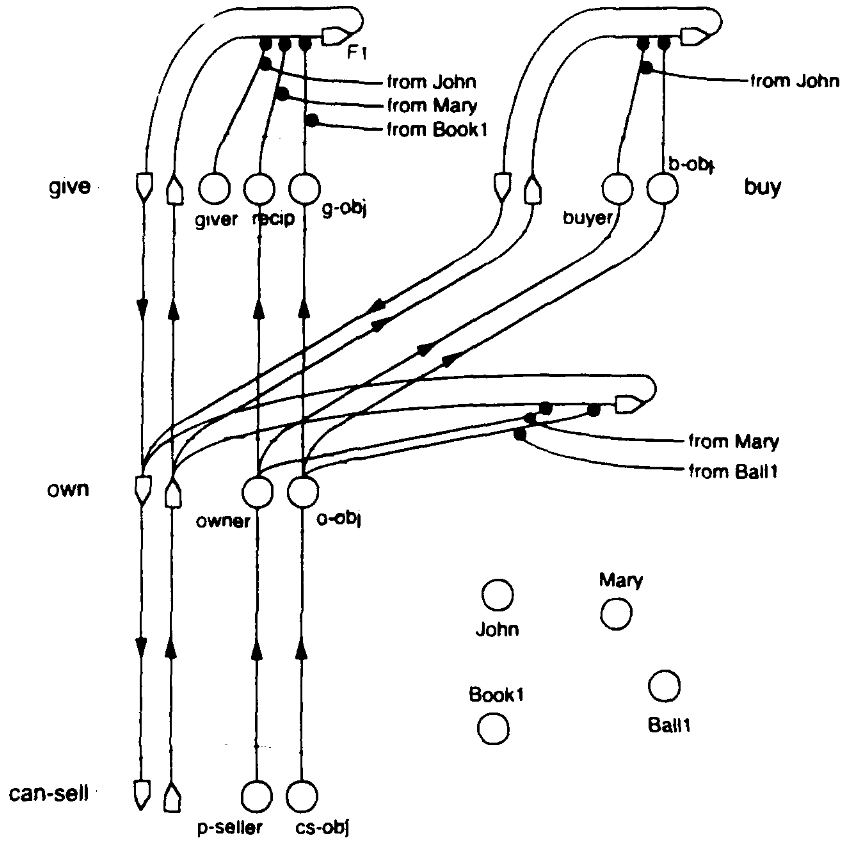


Figure 1. (a) An example encoding of rules and facts. The network encodes the following rules and facts: ∀x,y,z give(x,y,z) ⇒ own(y,z), ∀x,y buy (x,y) ⇒ own(x,y), ∀x,y own(x,y) ⇒ can-sell(x,y), give(John,Mary,Book1), buy(John,x) and own(Mary,Ball1).

Throughout the paper, we will mostly concern ourselves with backward reasoning, unless explicitly stated otherwise.

2. The Basic Rule-based Reasoning System

A brief description of the basic reasoning system is provided here. The reader is referred to Shastri and Ajjanagadde (1990) for a detailed exposition of the reasoning system and its characteristics. Figure 1(a) illustrates how long-term knowledge is encoded in the rule-based reasoning system. The network shown in Figure 1(a) encodes the following rules and facts:

∀x,y,z give(x,y,z) ⇒ own(y,z)

∀x,y buy(x,y) ⇒ own(x,y)

∀x,y own(x,y) ⇒ can-sell(x,y)

give(John,Mary,Book1)

buy(John,x)

own(Mary,Ball1).

(b)

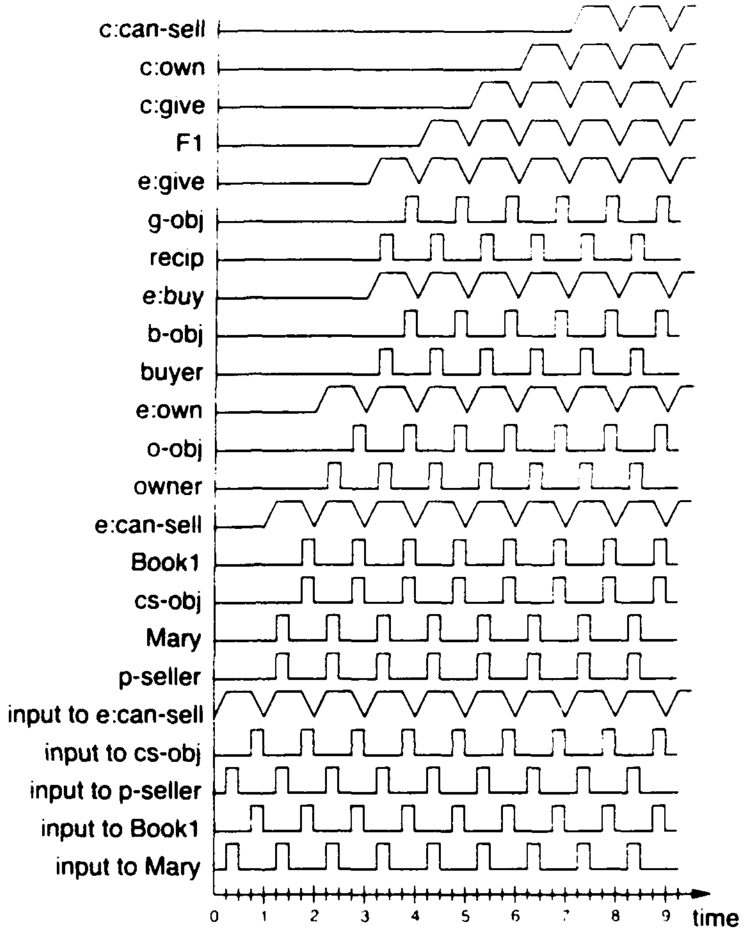


Figure 1. (b) Activation trace for the query can-sell(Mary,Book1)? ('Can Mary sell Book1?'). The inputs (to nodes e:can-sell, cs-obj, p-seller, Book1 and Mary) needed to pose the query are also shown. In other activation trace diagrams to follow, these inputs will not be explicitly shown.

The rule ∀x,y,z give(x,y,z) ⇒ own(y,z) states that 'if x gives z to y, then y owns z'. The other two rules are interpreted similarly. The facts give(John,Mary,Book1) and own(Mary,Ball1) represent 'John gave Mary Book1' and 'Mary bought Ball1', respectively, while buy(John,x) states that 'John bought *something*'.

The encoding of rules and facts makes use of several types of nodes (see Figure 2): ρ-btu nodes (depicted as circles), τ-and nodes (depicted as pentagons) and τ-or nodes (depicted as triangles). These nodes have the following idealized behavior. If ρ-btu node A is connected to another ρ-btu node B, then the activity of node B will synchronize with the activity of node A. In particular, a periodic firing of A will lead to a periodic and in-phase firing of B.[4] We assume that ρ-btu nodes can respond in this manner as long as the period of firing, π, lies in the interval $[\pi_{min}, \pi_{max}]$. This interval can be interpreted as defining the frequency range over which ρ-btu nodes can sustain a synchronized response. For a discussion of biologically motivated values for these parameters, see Shastri and

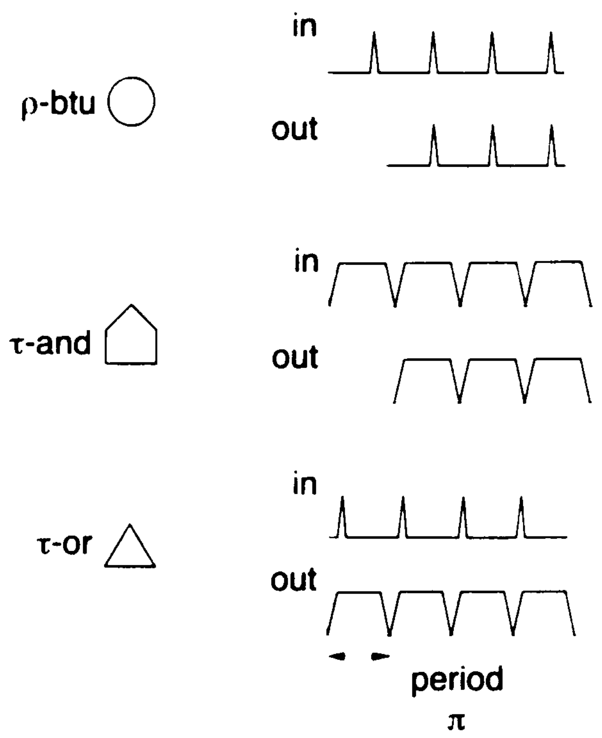


Figure 2. Input–output behavior for the ρ-btu, τ-and and τ-or nodes used in the reasoning system. ρ-btu ( ) and τ-and (⌂) nodes can fire with any period in the interval $[\pi_{\min}, \pi_{\max}]$. τ-or (△) nodes always fire with period $\pi_{\max}$.

Ajjanagadde (1993a). A τ-and node behaves like a temporal AND node, and becomes active on receiving an uninterrupted pulse train. On becoming active, a τ-and node produces a pulse train similar to the input pulse train. A τ-or node, on the other hand, becomes active on receiving any activation; its output is a pulse whose width and period equal $\pi_{\max}$. Figure 2 summarizes the behavior of these nodes for the idealized case of oscillatory inputs.

The maximum number of distinct entities that may participate in an episode of reasoning equals $\lfloor \pi/\omega \rfloor$ where π is the period of oscillation. We define ω to be the width of the window of synchronization—nodes firing with a lag or lead of less than $\omega/2$ would be considered to be in synchrony. The encoding also makes use of inhibitory modifiers—links that impinge upon and inhibit other links. A pulse propagating along an inhibitory modifier will block a pulse propagating along the link it impinges upon. In Figure 1(a), inhibitory modifiers are shown as links ending in solid circles.

Each entity in the domain is encoded by a ρ-btu node. An n-ary predicate P is encoded by a pair of τ-and nodes and n ρ-btu nodes, one for each of its n arguments. One of the τ-and nodes is referred to as the enabler, $e{:}P$, and the other as the collector, $c{:}P$. In Figure 1(a), enablers point upwards while collectors point downwards. The enabler $e{:}P$ becomes active whenever the system is being queried about P. On the other hand, the system activates the collector $c{:}P$ of a predicate P whenever the system wants to assert that the current dynamic bindings of the arguments of P follow from the knowledge encoded in the system.

A rule is encoded by connecting the collector of the antecedent predicate to the collector of the consequent predicate, the enabler of the consequent predicate to the enabler of the antecedent predicate, and by connecting the arguments of the consequent predicate to the arguments of the antecedent predicate in accordance with the correspondence between these arguments specified in the rule. A fact is encoded using a τ-and node that receives an input from the enabler of the associated predicate. This input is modified by inhibitory modifiers from the argument nodes of the associated predicate. If an argument is bound to an entity in the fact, then the modifier from such an argument node is in turn modified by an inhibitory modifier from the appropriate entity node. The output of the τ-and node is connected to the collector of the associated predicate. Figure 1(a) shows the encoding of the facts give(John,Mary,Book1), own(Mary,Ball1) and buy(John,x). Note that the fact buy(John,x) would be triggered only if the second argument of the buy predicate is unbound, i.e. the object which John bought is unspecified.

2.1. *The Inference Process*

Posing a query to the system involves specifying the query predicate and the argument bindings specified in the query. In the proposed system, this is done by simply activating the relevant nodes in the manner described below. Let us choose an arbitrary point in time—say, t_0—as our point of reference for initiating the query. We assume that the system is in a quiescent state just prior to t_0. The query predicate is specified by activating the enabler of the query predicate with a pulse train of width and periodicity π starting at time t_0.

The argument bindings specified in the query are communicated to the network as follows. Let the argument bindings in the query involve n distinct entities: $c_1, \ldots, c_n$. With each c_i, associate a delay δ_i such that no two delays are within ω of one another and the longest delay is less than $\pi - \omega$. As mentioned earlier, ω is the width of the window of synchrony and π is the period of oscillation. Each of these delays may be viewed as a distinct phase within the period t_0 and $t_0 + \pi$. Now the argument bindings of an entity c_i are indicated to the system by providing an oscillatory spike train of periodicity π starting at $t_0 + \delta_i$, to c_i and all arguments to which c_i is bound. This is done for each entity c_i ($1 \leq i \leq n$) and amounts to representing argument bindings by the in-phase or synchronous activation of the appropriate entity and argument nodes.

We illustrate the reasoning process with the help of an example. Consider the query can-sell(Mary,Book1)? (i.e. 'Can Mary sell Book1?') This query is posed by providing inputs to the entities Mary and Book1, the arguments p-seller and cs-obj, and the enabler e:can-sell, as shown in Figure 1(b). Mary and p-seller receive in-phase activation and so do Book1 and cs-obj. Let us refer to the phase of activation of Mary and Book1 as ρ_1 and ρ_2, respectively. As a result of these inputs, Mary and p-seller will fire synchronously in phase ρ_1 of every period of oscillation, while Book1 and cs-obj will fire synchronously in phase ρ_2 of every period of oscillation. The node e:can-sell will also oscillate and generate a pulse train of periodicity and pulse width π. The activations from the arguments p-seller and cs-obj reach the arguments owner and o-obj of the own predicate, and, consequently, starting with the second period of oscillation, owner and o-obj become active in ρ_1 and ρ_2, respectively. At the same time, the activation from e:can-sell activates e:own. The system has, essentially, created dynamic bindings for the arguments of predicate own. Mary has been bound to the argument owner, and Book1 has been bound to the argument o-obj. These newly created bindings in conjunction with the activation of e:own can be thought of as encoding the query own(Mary,Book1)? (i.e. 'Does Mary own Book1?'). The τ-and node associated with the fact own(Mary,Ball1) does not match the query and remains inactive. The activations from owner and o-obj reach the arguments recip and g-obj of give, and buyer and b-obj of buy, respectively. Thus, beginning with the third period of oscillation, arguments recip and buyer become active in ρ_1, while arguments g-obj and b-obj become active in ρ_2. In essence, the system has created new bindings for the predicates give and buy that can be thought of as encoding two new queries: give(x,Mary,Book1)? (i.e. 'Did *someone* give Mary Book1?') and buy(Mary,Book1)? Observe that now the τ-and node associated with the fact give(John,Mary,Book1)—this is the τ-and node labeled *F*1 in Figure 1(a)—becomes active as a result of uninterrupted activation from e:give. The inhibitory inputs from recip and g-obj are blocked by the in-phase inputs from Mary and Book1, respectively. The activation from this τ-and node causes c:give, the collector of give, to become active. The output from c:give in turn causes c:own to become active and transmit an output to c:can-sell. Consequently, c:can-sell, the collector of the query predicate can-sell, becomes active (refer to Figure 1(b)) resulting in an affirmative answer to the query can-sell(Mary,Book1)?

3. The Type Hierarchy

3.1. Overview

Figure 3(a) gives an overview of the reasoning system augmented with a type hierarchy. The rule-based part of the network encodes the rule ∀x,y preys-on(x,y) ⇒ scared-of(y,x) (i.e. 'if *x* preys on *y*, then *y* is scared of *x*'), and the facts ∀x:Cat, y:Bird preys-on(x,y) and ∃x:Cat ∀y:Bird loves(x,y). The former fact is equivalent to preys-on(Cat,Bird) and the amounts to 'cats prey on birds'. The latter amounts to 'there is a cat that loves all birds'. The network on the right in Figure 3(a) encodes the following is-a relationships: is-a(Bird,Animal), is-a(Cat,Animal), is-a(Robin,Bird), is-a(Canary,Bird), is-a(Chirpy,Robin), is-a(Tweety,Canary) and is-a(Sylvester,Cat).

3.1.1. Interpreting facts. Facts involving typed variables are interpreted in the following manner:

- A typed, universally quantified variable is treated as being equivalent to its type. Also, any entity directly specified in a fact is treated as a substitute for a typed universal variable.[5] Thus, ∀x:Cat, y:Bird preys-on(x,y), ∀x:Cat preys-on(x,Bird) and ∀y:Bird preys-on(Cat,y) are all encoded as preys-on(Cat,Bird).
- A typed, existentially quantified variable is encoded using a unique sub-concept of the associated type. Thus, in Figure 3(a), ∃x:Cat ∀y:Bird loves(x,y) is encoded as loves(Cat-1,Bird), where Cat-1 is assumed to be a unique instance of Cat.[6]

The example in Section 3.4 clarifies these notions. Note that this scheme deals only with existential variables outside the scope of universally quantified variables.

(a)

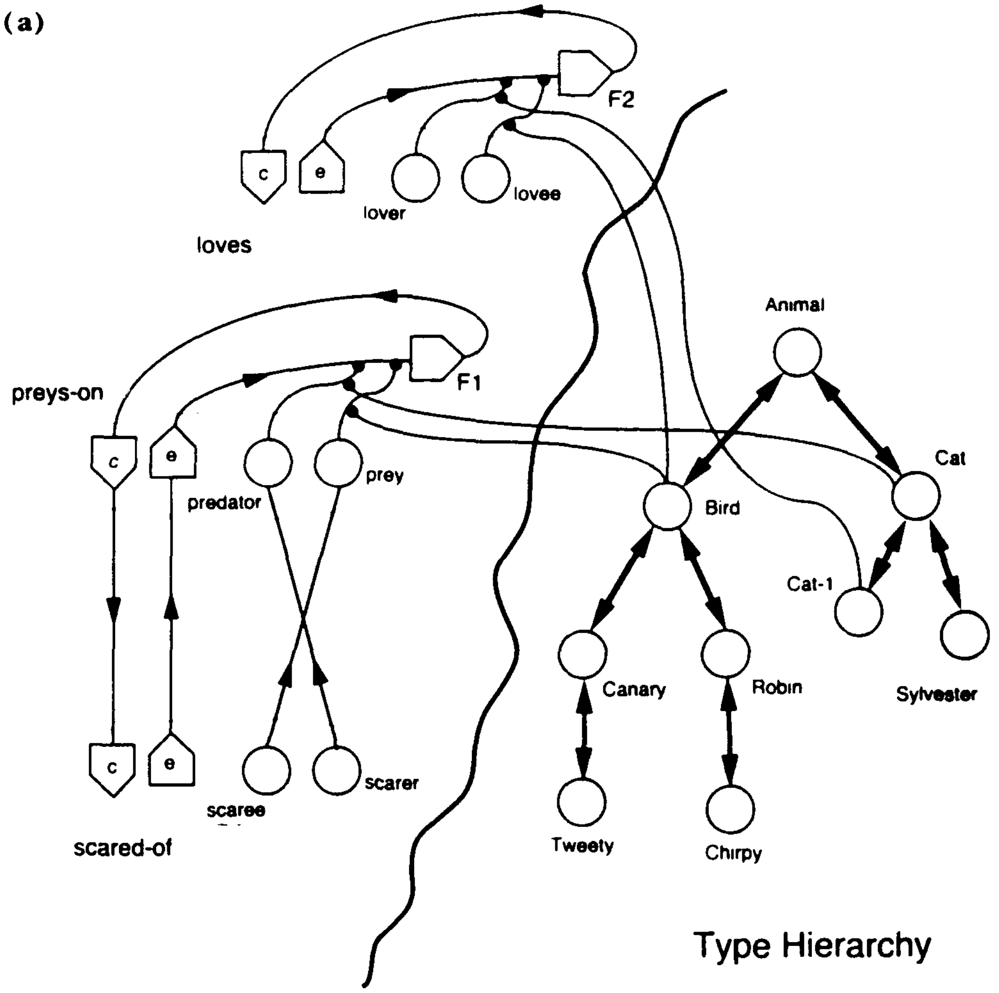


Figure 3. (a) Interaction between the reasoner and the type hierarchy. The rule base encodes the rule ∀x,y preys-on(x,y) ⇒ scared-of(y,x) and the facts ∀x:Cat, y:Bird preys-on(x,y) and ∃x:Cat ∀y:Bird loves(x,y). The type hierarchy encodes the following is-a relations: is-a(Bird,Animal), is-a(Cat,Animal), is-a(Robin,Bird), is-a(Canary,Bird), is-a(Chirpy,Robin), is-a(Tweety,Canary) and is-a(Sylvester,Cat).

For now, let us assume the following: each type or instance is encoded as a ρ-btu node; each conceptual is-a relationship such as is-a(A,B) is encoded using two connectionist links—a bottom-up link from A to B and a top-down link from B to A; and that the top-down and bottom-up links can be enabled selectively by built-in control mechanisms.

The time course of activation for the query scared-of(Tweety,Sylvester)? ("Is Tweety scared of Sylvester?") is given in Figure 3(b). The query is posed by turning on e:scared-of and activating the nodes Tweety and Sylvester in synchrony with the first (scaree) and second (scarer) arguments of scared-of, respectively. The bottom-up links emanating from Tweety and Sylvester are also enabled. In the rule base, activation from scaree spreads to prey, the second

(b)

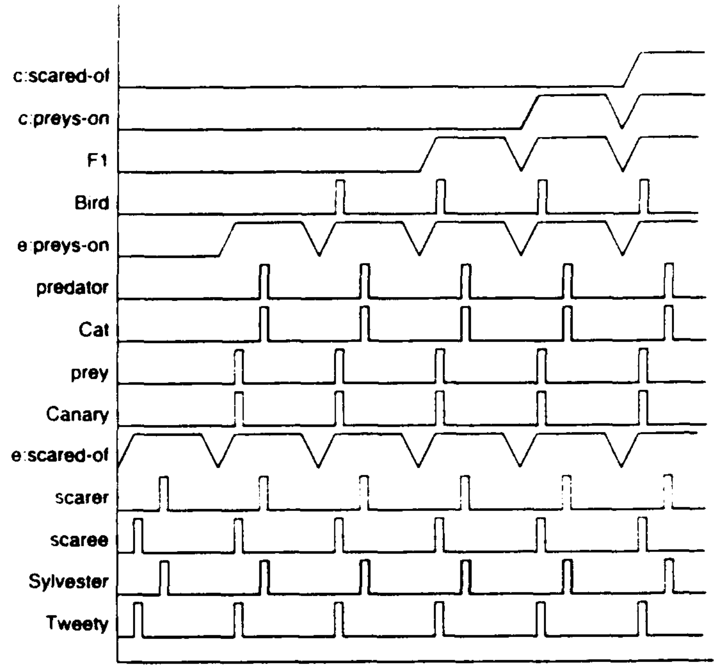


Figure 3. (b) Trace of spreading activation for the query scared-of(Tweety,Sylvester)? ('Is Tweety scared of Sylvester?').

argument of the preys-on predicate. Similarly, the activation of scarer spreads to predator. At the same time, e:scared-of activates e:preys-on. As a result, the initial query scared-of(Tweety,Sylvester)? is rephrased as preys-on(Sylvester,Tweety)? ('Does Sylvester prey on Tweety?'). Concurrently with activation spread in the rule base, activation also propagates in the type hierarchy. This causes Canary and Bird to fire in synchrony with Tweety, and Cat in synchrony with Sylvester. The net result of activation propagation in the rule base and type hierarchy is to transform the query scared-of(Tweety,Sylvester)? into the query preys-on(Cat,Bird)? (refer to Figure 3(b)). The latter query matches the stored fact preys-on(Cat,Bird) and leads to the activation of c:preys-on. In turn, c:scared-of becomes active and signals an affirmative answer to the query.

3.2. Two Technical Problems in Realizing a Type Hierarchy

There are two technical problems that must be solved in order to integrate the type hierarchy and the rule-based component.

First, the encoding of the is-a hierarchy should be capable of representing multiple instantiations of a concept. For example, in the query discussed above, the concept Animal would receive activation originating from Tweety as well as Sylvester. We would like the network's state of activation to represent both 'the animal Tweety' and the 'the animal Sylvester' This is problematic because the ρ-btu node representing Animal cannot be in synchrony with both Tweety and Sylvester at the same time.

Second, the encoding must provide built-in mechanisms for controlling the propagation of activation in the is-a hierarchy so as to deal correctly with queries containing existentially and universally quantified variables. Thus,

- Activation originating from an instance or a concept C that corresponds to a universally quantified variable *in the query* should propagate upwards to all its ancestors. Activation propagating upwards is equivalent to checking if the relevant fact is universally true for some ancestor of C, in which case it is true for all C.
- If the is-a hierarchy is a taxonomy, then activation originating from a concept C that corresponds to an existentially quantified variable in the query should propagate to the ancestors as well as descendants of C. A fact is true for some object of type C if at least one of the following holds:
  - — The fact is universally true for an ancestor of C. Activation traveling upwards from C checks this case.
  - — The fact is true for some descendant of C. Activation traveling downwards from C is meant to check this condition.

  However, if the is-a hierarchy permits multiple inheritance, then the fact would be true of C if it is universally true for an ancestor of a descendant of C. This requires that the activation must also propagate to the ancestors of the descendants of C. The multiple inheritance situation is illustrated by the scenario in Figure 4: if 'all pets are lovable', then it also follows that 'there exists some animal that is lovable'. Given the fact ∀x:Pet lovable(x) ('all pets are lovable'), to be able to give an affirmative answer to the query ∃x:Animal lovable(x)? ('Is there some animal that is lovable?'), we need to be able to propagate activation to all the ancestors of the descendants of Animal. Thus, we require activation originating from a concept C, which corresponds to an existentially quantified variable, to propagate to its ancestors, descendants and ancestors of descendants. Again, the example in Section 3.4 clarifies these notions.

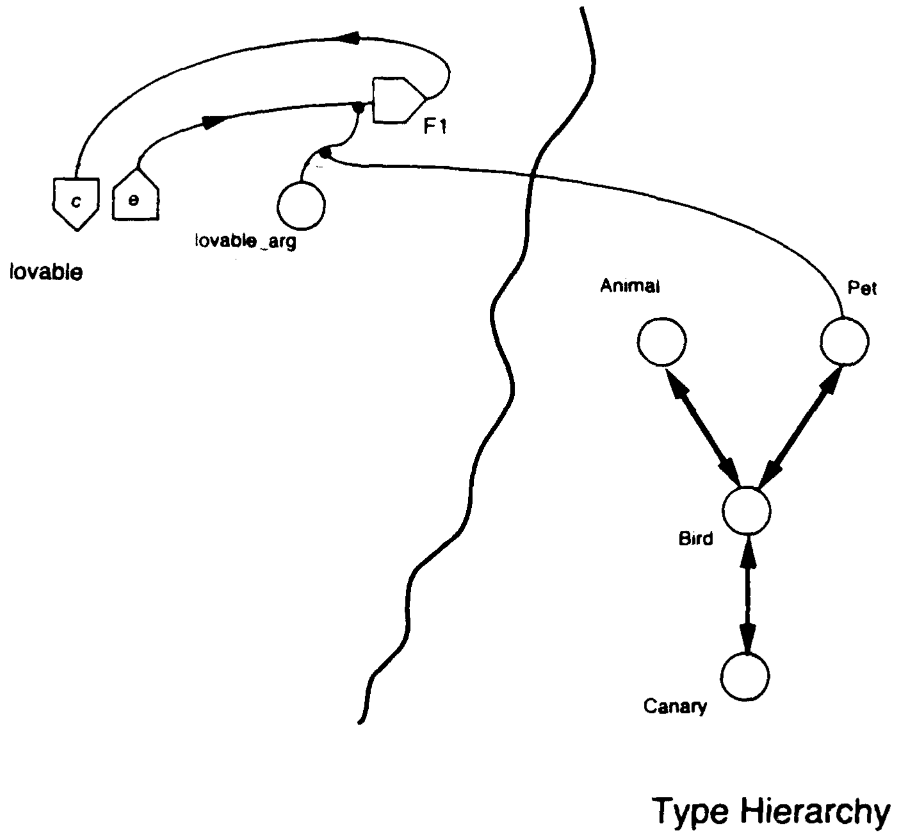
The next section proposes a solution to these problems.

3.3. Implementing the Type Hierarchy

3.3.1. Representing entities. Each entity (i.e. type or instance) C is represented by a group of nodes called the entity cluster for C. Such a cluster is organized as shown in Figure 5(a). The entity cluster for C has k_1 banks of ρ-btu nodes, where k_1, the type hierarchy multiple instantiation constant, refers to the number of dynamic instantiations a concept can accommodate. Each bank C_{B_i} consists of three ρ-btu nodes: C_i, $C_{i\uparrow}$, $C_{i\downarrow}$. Each C_i represents a distinct (dynamic) instantiation of C. If this instantiation is in phase ρ, then C_i fires in phase ρ. The relay nodes $C_{i\uparrow}$ and $C_{i\downarrow}$ control the direction of propagation of the activation represented by C_i. The $C_{i\uparrow}$ and $C_{i\downarrow}$ nodes have a threshold $\theta = 2$. As shown in Figure 5(a), C_i is connected to both $C_{i\uparrow}$ and $C_{i\downarrow}$. $C_{i\downarrow}$ is linked to $C_{i\uparrow}$, but not vice versa. Directional control of propagating activation is exercised using a suitable modification of the relay-node scheme discussed in Shastri (1988).

3.3.2. The type hierarchy switch (T-switch). Every entity C is associated with two type hierarchy switches—a top-down T-switch and a bottom-up T-switch. In order to avoid confusion with switches introduced to handle multiple dynamic

(a)



(b)

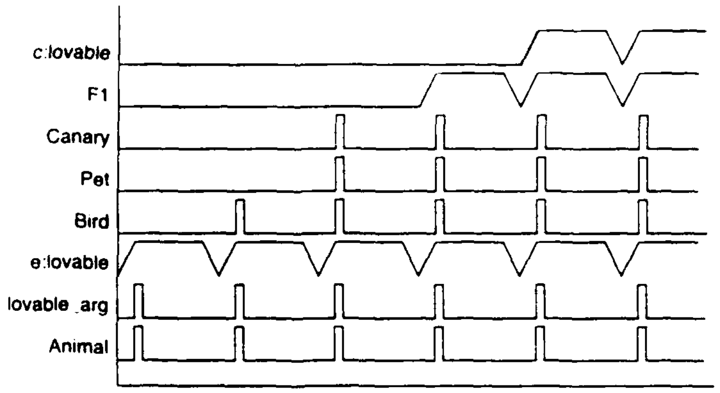


Figure 4. (a) Fragment of a type hierarchy with multiple inheritance, encoding the fact ∀x:Pet lovable(x). The type hierarchy encodes the following is-a relations: is-a(Bird,Animal), is-a(Bird,Pet) and is-a(Canary,Bird). (b) Activation trace for the query ∃x:Animal lovable(x)? ("Is there an animal that is lovable?").

instantiations of predicates, we shall refer to the switches used in the type hierarchy as T-switches. The T-switches, both of which are identical in structure, control the flow of activation in the type hierarchy. Each T-switch has k_1 outputs. *Output*$_i$ from the bottom-up T-switch connects to C_i and $C_{i\uparrow}$ while the corresponding output from the top-down T-switch goes to the C_i and $C_{i\downarrow}$ nodes,

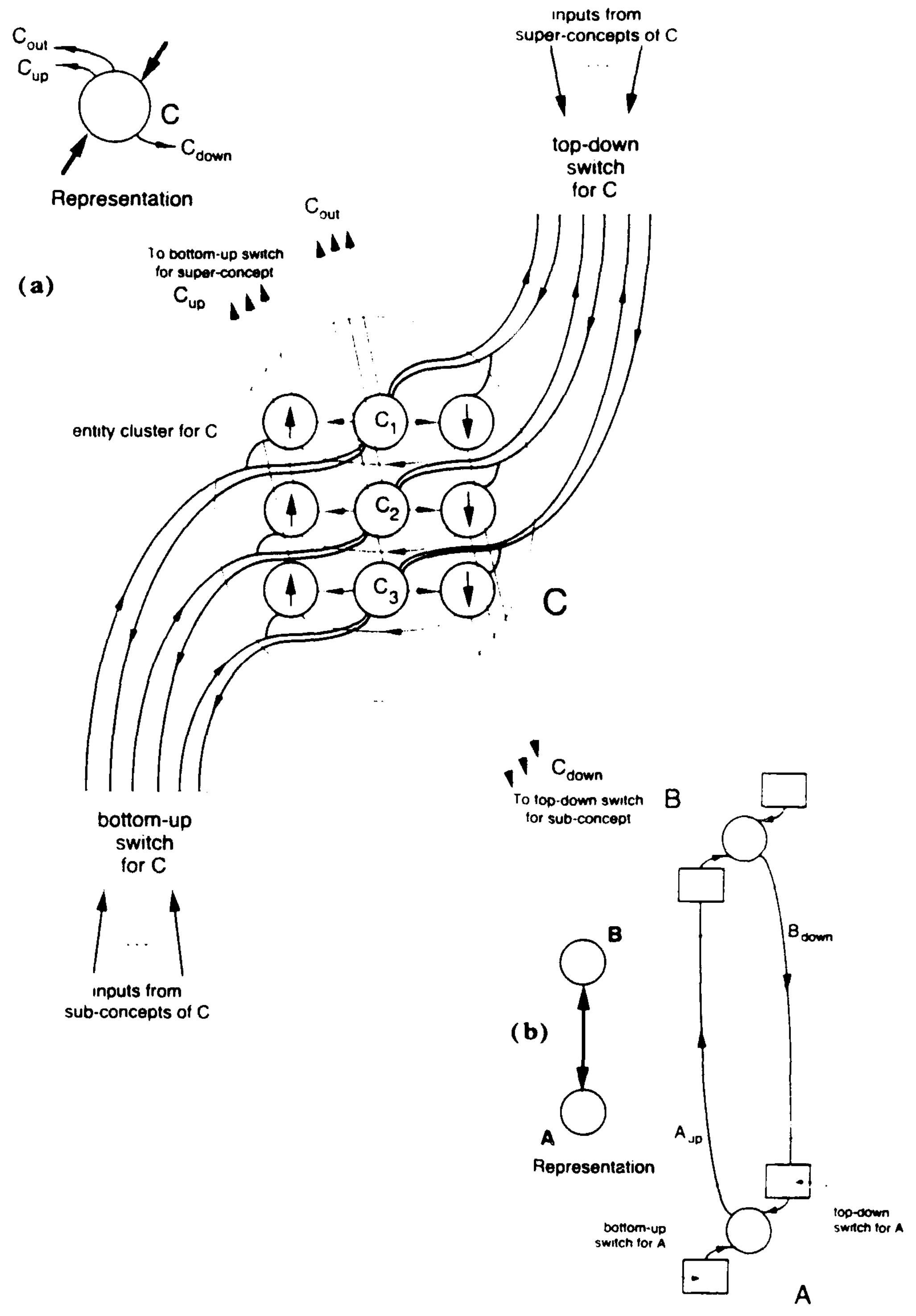


Figure 5. (a) Structure of the entity cluster for C, and its interaction with the bottom-up and top-down type hierarchy switches. The ↑ and ↓ nodes have a threshold $\theta = 2$. The multiple instantiation constant, $k_1 = 3$, and represents the number of instantiations that can be represented in any entity cluster. (b) Encoding of the is-a relation is-a(A,B). A bundle of k_1 wires is represented by a single link.

$1 \leq i \leq k_1$. The bottom-up T-switch has $k_1 \cdot n_{sub}$ inputs while the top-down T-switch has $k_1 \cdot n_{sup}$ inputs, n_{sub} and n_{sup} being the number of sub- and super-concepts of C, respectively. Further, there is also a feedback from the C_i nodes to both the T-switches (see Figure 5(a) and Figure 6).

The interaction between the T-switches and the entity cluster (Figure 5(a)) brings about efficient and automatic dynamic allocation of banks in an entity cluster, by ensuring that:

- Activation is channeled to the entity cluster banks only if the entity cluster can accommodate more instantiations; the maximum number of instantiations is, therefore, limited to k_1.
- Each C_i picks up a unique phase; thus, new instantiations are always in a phase not already represented in the entity cluster.

The architecture of the T-switch (with $k_1 = 3$) is illustrated in Figure 6. The k_1 ρ-btu nodes, $S_1, \ldots, S_{k_1}$, with their associated τ-or nodes form the basic components of the T-switch. Every input to the T-switch makes two connections—one excitatory and one inhibitory—to each of $S_2, \ldots, S_{k_1}$; these inputs directly connect to S_1. As a result of these excitatory-inhibitory connections, the $S_2, \ldots, S_{k_1}$ nodes cannot respond to incoming activation. Input activation will

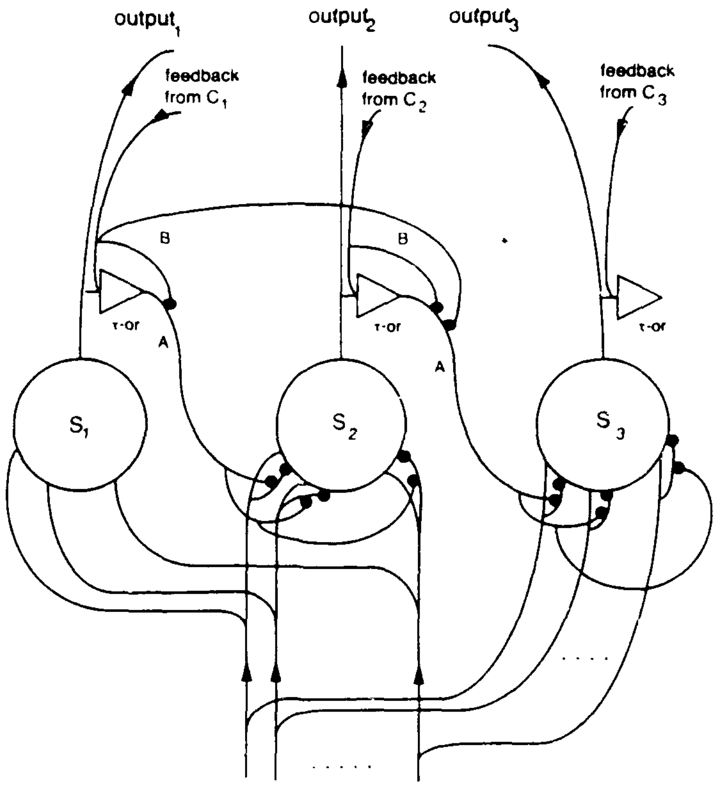


Figure 6. Architecture of the type hierarchy T-switch, which arbitrates the flow of activation in the type hierarchy. The multiple instantiation constant $k_1 = 3$, and represents the number of instantiations that can be represented in any entity cluster.

therefore have an effect only on the S_1 node (Figure 6). In keeping with the behavior of ρ-btu nodes (Section 2), S_1 selects an arbitrary active input and continues to fire in phase with that input as long as it remains active. As S_1 goes active, the τ-or node associated with S_1 turns ON, thereby enabling S_2 (via the A link). Inhibitory feedback from C_1 (via the B link) ensures that S_2 is not enabled during the phase ρ in which C_1 is firing. Thus, S_2 selects and starts firing in a phase other than ρ. Once S_2 has made its selection, S_3 gets its turn, and so on.

Note that, in general, C_i could receive input in two phases—one from the bottom-up T-switch for C and another from its top-down T-switch. C_i, being a ρ-btu node, picks one of these phases to fire in. As instantiations are deputed to the entity cluster, the ρ-btu nodes in the T-switch are progressively enabled from left to right. If $C_1, \ldots, C_{i-1}$ are firing in phases $\rho_1, \ldots, \rho_{i-1}$, then S_i always picks a distinct phase $\rho \notin \{\rho_1, \ldots, \rho_{i-1}\}$, since inputs in phases $\rho_1, \ldots, \rho_{i-1}$ are inhibited by the feedback links from $C_1, \ldots, C_{i-1}$. At any stage, if C_i, $1 \leq i \leq k_1$ picks up activation channeled by the other T-switch, feedback from C_i into the τ-or node associated with S_i causes S_{i+1} to be enabled, even though S_i has not picked a phase. This mechanism ensures that at most k_1 instantiations are selected jointly by the bottom-up and top-down T-switches; hence, only k_1 instantiations can be channeled to C, at worst.

3.3.3. Connecting up the type hierarchy. A fact of the form is-a(A,B) is represented as shown in Figure 5(b) by connecting the $A_{i\uparrow}$, $i = 1, \ldots, k_1$ nodes to the bottom-up T-switch for B; and connecting the $B_{i\downarrow}$, $i = 1, \ldots, k_1$ nodes to the top-down T-switch for A.

Consider a concept C in the type hierarchy. Suppose C_i receives activation from the bottom-up T-switch in phase ρ. C_i starts firing in synchrony with this activation. The $C_{i\uparrow}$ node is now receiving two inputs in phase ρ (one from the bottom-up T-switch and the other from C_i; see Figure 5(a)). Since it has a threshold $\theta = 2$, $C_{i\uparrow}$ also fires in phase ρ. This causes activation in phase ρ to spread eventually to the super-concept of C. Hence, any upward traveling activation continues to travel upwards—which is the required behavior when C is associated with a universal typed variable (Section 3.2). Similarly, when C_i receives activation from the top-down T-switch in phase ρ, both C_i and $C_{i\downarrow}$ become active in phase ρ. $C_{i\uparrow}$ follows suit, because of the link from $C_{i\downarrow}$ to $C_{i\uparrow}$, so that the whole bank C_{B_i} now fires in phase ρ. Thus, while any activation traveling downwards continues to travel downwards, it also sets off upward activation trails from every concept encountered on its way. This mechanism allows a concept associated with an existential typed variable to spread its activation eventually to its ancestors, descendants and ancestors of descendants, which is in keeping with the desired behavior mentioned in Section 3.2. Note that the behavior of downward activation is unlike that of upward activation—upward activation just continues upwards while downward activation, apart from continuing downwards, also sets off an upward trail.

3.4. *Example*

Assuming that each concept in the type hierarchy shown in Figure 3(a) has the structure indicated in Figure 5(a), the query ∃x:Cat loves(x,Tweety)? ('Is there a cat that loves Tweety?') would be posed by: (i) activating Cat_1 and $\text{Cat}_{1\downarrow}$ to fire in synchrony with **lover**; (ii) activating **Tweety$_1$** and **Tweety$_{1\uparrow}$** to fire in synchrony with **lovee**; and (iii) activating **e:loves**, the enabler of the **loves** predicate.

Since **Cat$_1$** is associated with an existential variable, **Cat$_1$** and **Cat$_{1\downarrow}$** are activated (see Section 5.2) and this activation spreads to **Animal$_1$** (upwards) and to **Cat-1$_1$** (downwards). Activation from **Cat$_1$** also spreads to the ancestors of its descendants. Since the ancestors of the descendants of **Cat** are **Cat** and **Animal**, and since these concepts already have banks firing in phase with **Cat$_1$**, no new instantiations are introduced. **Tweety** is an entity directly appearing in the query and is equivalent to a universally typed variable. Activation from **Tweety$_1$** therefore only propagates upwards, to **Canary$_1$**, **Bird$_1$** and **Animal$_2$**. Figure 7 shows the resulting spread of activation in the network. The activity of the corresponding ↑ and ↓ nodes are also indicated.

Activation spreading downwards from **Cat$_1$** causes **Cat-1$_1$** to go active, while upward activation from **Tweety$_1$** eventually reaches **Bird$_1$**. When this happens,

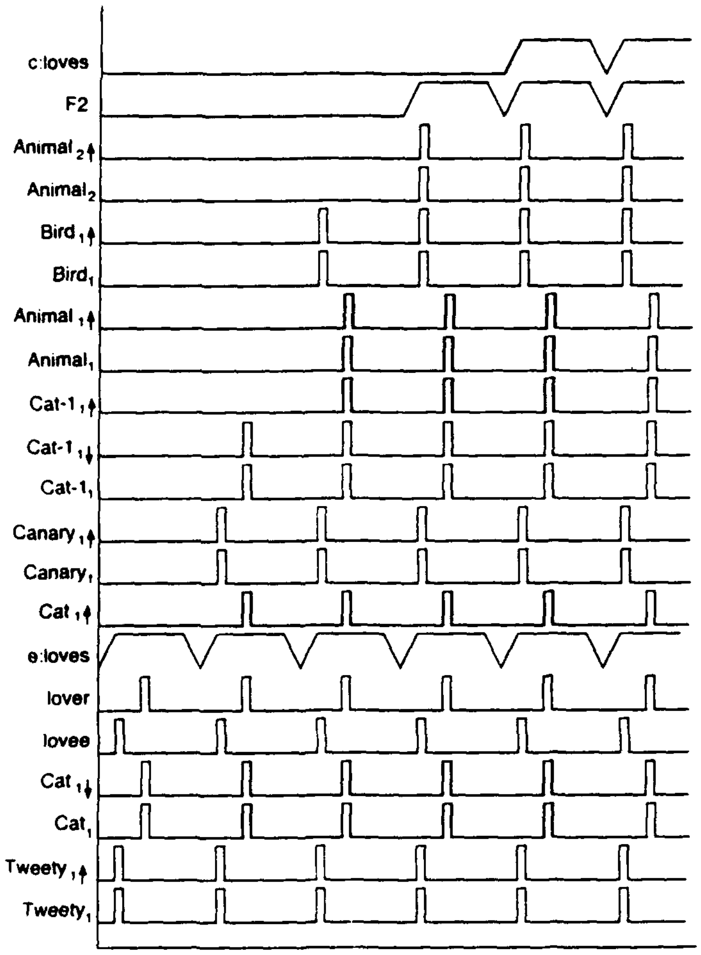


Figure 7. Trace of spreading activation for the query ∃x:Cat loves(x,Tweety)?. The rule base and the type hierarchy are as shown in Figure 3(a), except that the entities in the type hierarchy are assumed to have the structure indicated in Figure 5. Activation of only those nodes relevant to the query are shown. The time taken for activation to traverse the T-switches have been ignored in order to simplify the diagram.

the fact node $F2$ corresponding to the fact ∃x:Cat, ∀y:Bird loves(x,y) turns ON. This activates the enabler e:loves resulting in an affirmative answer to the query.

4. Multiple Dynamic Instantiation of Predicates

4.1. Overview

As mentioned in Section 1.2, being able to represent multiple dynamic facts about the same predicate provides several additional capabilities not possible in the original reasoner. Introduction of multiple instantiation relies on the assumption that, during an episode of reflexive reasoning, any given predicate need only be instantiated a bounded number of times. In Shastri and Ajjanagadde (1993a), it is argued that a reasonable value for this bound is around three. We shall refer to this bound as the multiple dynamic instantiation constant for predicates, k_2.

4.2. Implementing Multiple Dynamic Instantiation

4.2.1. Representing predicates. Since every predicate must now be capable of representing up to k_2 dynamic instantiations, predicates are represented using k_2 banks of units. Each bank of an n-ary predicate P consists of τ-and nodes for the collector ($c{:}P$) and enabler ($e{:}P$) along with n ρ-btu nodes ($P_{\mathrm{arg}_1}, \ldots, P_{\mathrm{arg}_n}$) representing the arguments of P. Each bank is essentially similar to the predicate representation used in Shastri and Ajjanagadde (1990). Figure 8 illustrates the structure of predicates in the system. Note that the enabler, $e{:}P$, and the arguments, $P_{\mathrm{arg}_1}, \ldots, P_{\mathrm{arg}_n}$, have a threshold $\theta = 2$.

For a given predicate P, the enabler of the ith bank $e{:}P_i$ will be active whenever the ith bank has been instantiated with some dynamic binding. The collector $c{:}P_i$ of the ith bank will be activated whenever the dynamic bindings in the ith bank follow from the knowledge encoded in the system.

4.2.2. The multiple instantiation switch (M-switch). Every predicate in the extended system has an associated multiple instantiation switch, referred to as the M-switch.[8] All connections to a predicate are made through its M-switch. The M-switch has k_2 output cables (see Figure 8), each of which connects to one bank of the predicate. A cable is a group of wires originating or terminating at a predicate bank; a cable, therefore, has wires from all the units (collector, enabler and argument units) in a bank. Each output cable from the M-switch is accompanied by a latch enable link. Activation of the latch enable link associated with the ith output cable indicates that the M-switch has successfully selected an instantiation for the ith bank of the predicate.

The M-switch arbitrates input instantiations to its associated predicate and brings about efficient and automatic dynamic allocation of predicate banks by ensuring the following:

- Fresh predicate instantiations are channeled to the predicate banks only if the predicate can accommodate more instantiations.
- All inputs that transform to the same instantiation are mapped into the same predicate bank. Thus, new instantiations selected for representation in the predicate are always unique.

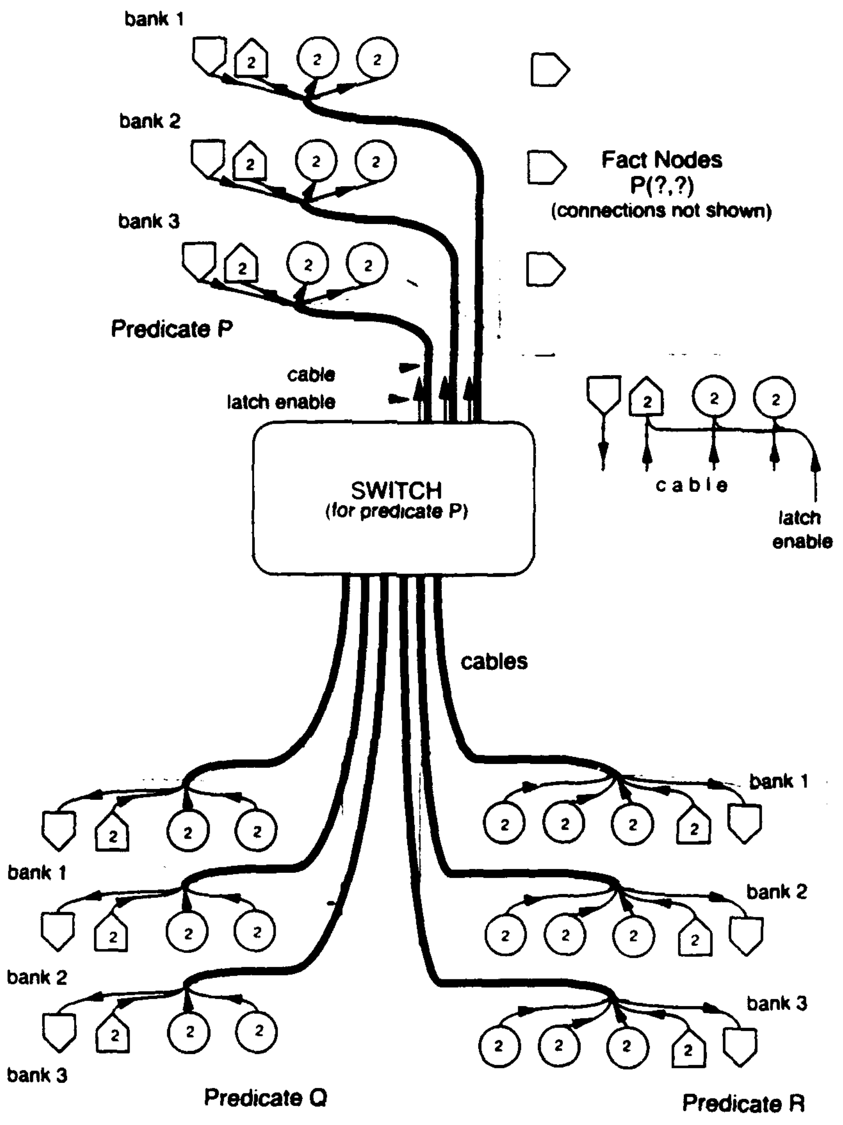


Figure 8. An overview of the multiple instantiation system. P and Q are binary predicates while R is a ternary predicate. Overall connection pattern depicting the encoding of two rules—one relating P and Q and the other relating P and R—is shown. Nodes marked with a '2' have a threshold $\theta = 2$. The multiple instantiation constant $k_2 = 3$, and represents the number of instantiations that can be simultaneously represented in a predicate.

4.2.3. Structure and operation of the multiple instantiation switch. Figure 9 illustrates the construction of the M-switch. The M-switch consists of k_2 groups or ensembles of units. The figures use $k_2 = 3$. The output of the ith ensemble is a cable (along with its latch enable link) which connects to the ith bank of the corresponding predicate.

As can be seen in Figure 9(a), each ensemble consists of an arbitrator bank and several input banks. The arbitrator consists of n ρ-btu nodes representing the

(a)

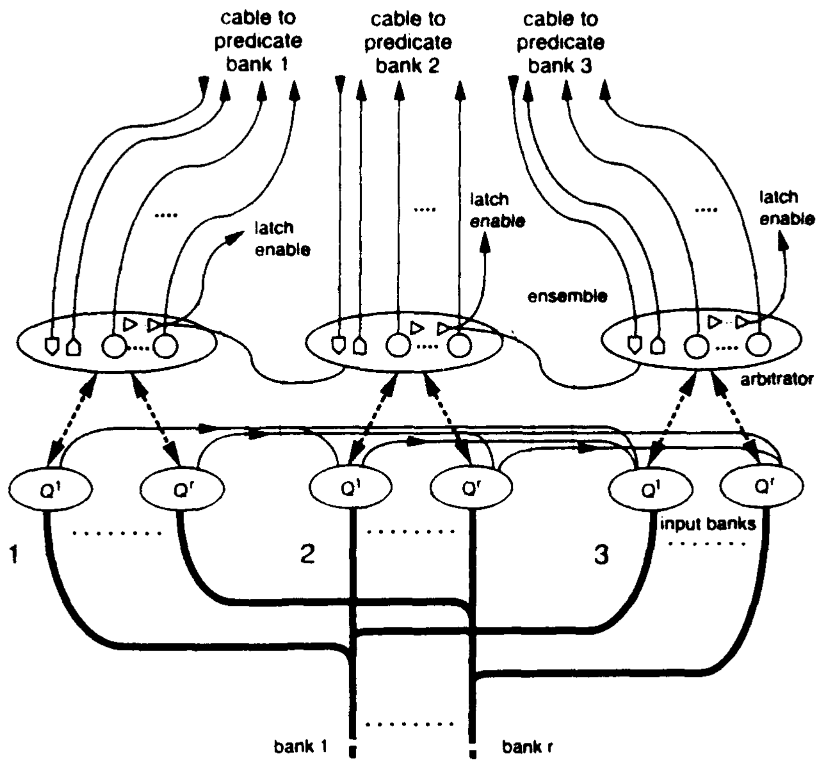


Figure 9. (a) Structure of the multiple instantiation M-switch which arbitrates the flow of dynamic instantiations into predicates. We have again assumed that the multiple instantiation constant $k_2 = 3$; this represents the number of instantiations that can be simultaneously represented in a predicate. Detailed connections are not shown to avoid cluttering.

arguments of the associated n-ary predicate, $n - 1$ τ-or nodes and two τ-and nodes for the collector and enabler. Each ρ-btu node, except for the node representing the first argument,[9] is associated with a τ-or node, as shown in Figure 9(b). The ith arbitrator bank directly connects with the ith bank of the predicate. Figure 9(b) shows the detailed structure of the arbitrator and input banks. Each input bank consists of n ρ-btu units representing the arguments of the predicate, and two τ-and nodes representing the collector and enabler of the bank. Each input bank also has a τ-or node associated with it. The cable terminating at the input bank is an input to the M-switch; the outputs of the input bank connect to the arbitrator of the respective ensemble. Corresponding input banks across ensembles are interconnected as shown in Figure 9.

Ignoring the associated τ-or nodes, the input banks and the arbitrators have a structure which exactly mimics the bank structure of the predicate with which the M-switch is associated. If the predicate has n arguments, the input banks and arbitrator banks also have n ρ-btu units. The number of lines in the input cable is decided by the arity of the predicate originating the cable. The number of lines in the M-switch output depends on the arity of the predicate associated with the M-switch. Since each input cable is connected to an input bank in each of the k_2 ensembles, each ensemble in the M-switch has the same number of input banks.

(b)

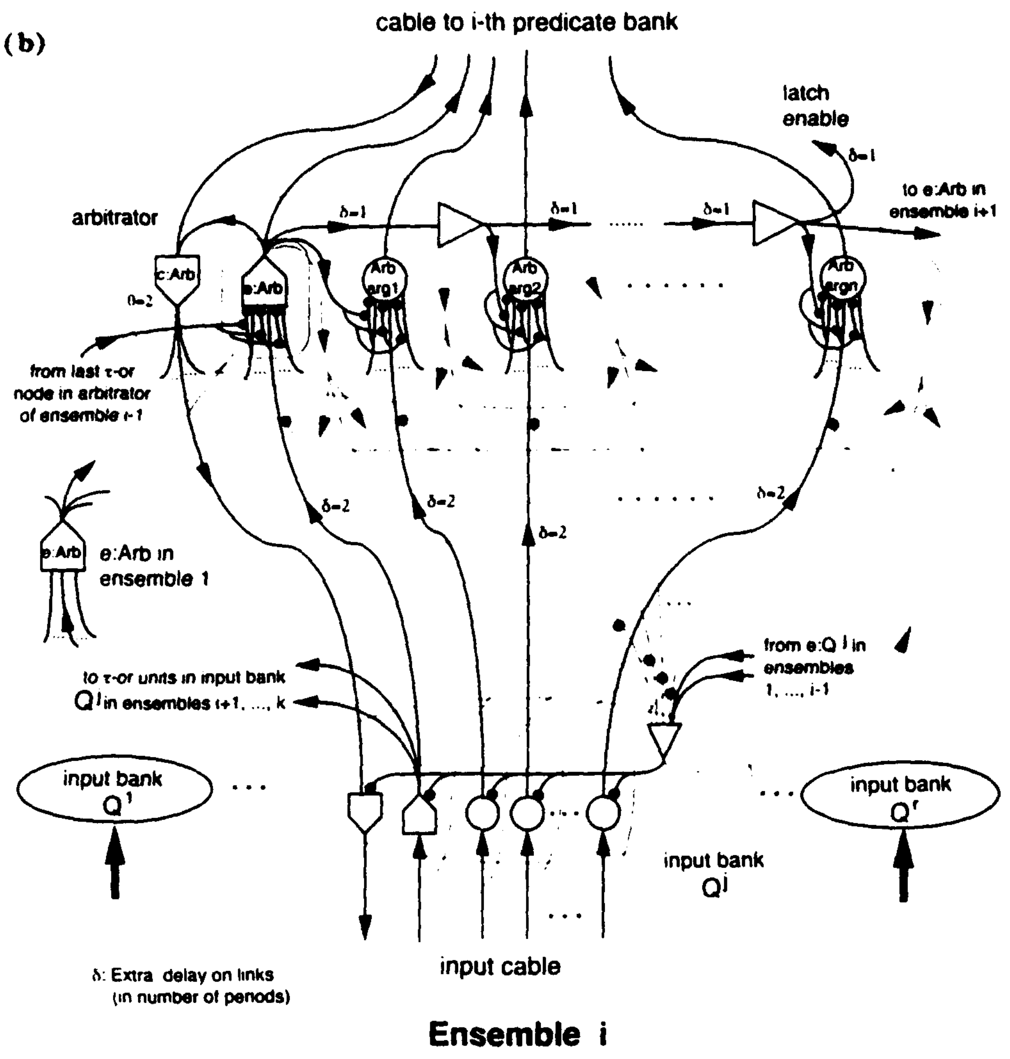

cable to i-th predicate bank
latch enable
δ=1
to e:Arb in ensemble i+1
arbitrator
c:Arb
e:Arb
Arb arg1
Arb arg2
Arb argn
δ=2
from last τ-or node in arbitrator of ensemble i-1
e:Arb in ensemble 1
to τ-or units in input bank Q^j in ensembles i+1, ..., k
from e:Q^j in ensembles 1, ..., i-1
input bank Q^1
input bank Q^j
input bank Q^r
input cable
δ: Extra delay on links (in number of periods)
Ensemble i

Figure 9. b) Structure of the *i*th ensemble in the M-switch. Only connections from input bank Q^j to the arbitrator are shown. Connections to/from other input banks and the arbitrator are implied. As indicated, connections to *e:Arb* in the first ensemble are different.

To start with, incoming instantiations will activate the corresponding input banks in all the ensembles of the switch. All ensembles in the switch except the first are disabled and cannot respond to incoming activation for the following reason: nodes in the arbitrators of all ensembles, except *e:Arb* in the first arbitrator, receive both an excitatory and an inhibitory input from their respective input units. The activation therefore cancels out and the arbitrator nodes in these other ensembles do not become active. Any activation incident on the switch will therefore affect only the first ensemble. Activation in one or more input banks of the first ensemble will cause the enabler in the arbitrator, *e:Arb*, to become active. All input banks with inactive enablers—i.e. input banks with no incoming activation—will be inhibited via the τ-or nodes associated with the respective input banks. The activation of *e:Arb* in the first ensemble will block the inhibitory inputs to the ρ-btu node, Arb_{arg_1}, thereby enabling this node to pick a phase to fire in. This node, in keeping with the behavior of ρ-btu nodes, arbitrarily selects one of its input phases and begins firing in that phase.[10] As soon as Arb_{arg_1} selects a phase to fire in, this phase is communicated to all the input banks, via the τ-or node associated with the input banks (see Figure 9(b)). For each of the input banks, the associated τ-or node checks if the phase selected by Arb_{arg_1} is the same as the phase in which the first argument of that input bank is firing in. If the phases do not match, the corresponding τ-or node shuts off the entire input bank. Thus, when Arb_{arg_1} selects a phase ρ from its input, all activation except that in which the first argument fires in phase ρ is inhibited.

In the meantime, *e:Arb* would have activated the τ-or node associated with the second argument, Arb_{arg_2}, in the arbitrator. This enables Arb_{arg_2} to select a phase from the activation remaining after inhibiting instantiations that do not agree with Arb_{arg_1}. Note that Arb_{arg_2} is enabled by the associated τ-or node independent of Arb_{arg_1} and will select a phase to fire in even if Arb_{arg_1} is inactive (which would be the case if all incoming instantiations have an unbound first argument). The process continues, allowing $Arb_{arg_3}, \ldots, Arb_{arg_n}$ to select phases to fire in. After, Arb_{arg_n} has made its choice, the first ensemble would have picked an instantiation to be channeled to the predicate bank. The latch enable, which originates at the τ-or node associated with Arb_{arg_n}, becomes active and the selected instantiation is transferred to the first predicate bank. A link from this last τ-or node to *e:Arb* in the second ensemble enables the second ensemble to select a fresh instantiation.

After the first ensemble has selected an instantiation to be channeled to the predicate, only those input banks which represent this exact pattern of activation will be active in the first ensemble. All other input banks will be inhibited due to a mismatch in the firing pattern. Further, input banks remaining active in the first ensemble will blot out activation in all corresponding input banks in all the other ensembles. This ensures that the instantiation selected by the first ensemble will not be selected again in any other ensemble.

Once the second ensemble is enabled (by blocking inhibitory inputs to *e:Arb* in the ensemble), it will pick an instantiation, channel it to the predicate bank, and enable the third ensemble in the M-switch, and so on. The process continues until k_2 instantiations have been channeled to the predicate, after which any fresh input instantiations are ignored.

Note that the ensembles in the M-switch have an implicit ordering from left to right (Figure 9(a)). If the ith ensemble ($1 \le i \le k_2$) is making its choice, it will always select an instantiation which is different from those picked by the first $i - 1$ ensembles. Further, a new instantiation arriving at the M-switch will be checked to see if it has already been assigned to a bank in the predicate. If so, the activation will be diverted to the bank already assigned to it.[11] If not, the activation is assigned a new bank in the predicate, via the next unused ensemble in the M-switch. Thus, all the instantiations channeled to the predicate are unique.

Further, whenever the collector of the ith bank of the predicate associated with the M-switch becomes active, the activation automatically gets transmitted to the collector of the predicate bank which originated the instantiation selected by the ith ensemble. Also note that *c:Arb* becomes active only if both *c:Arb* and the collector of the associated predicate bank are simultaneously active. A more detailed description of the structure and operation of the M-switch can be found in Mani and Shastri (1992).

5. Encoding Rules and Facts

Having described the mechanisms used to encode the type hierarchy and implement multiple dynamic instantiation of predicates, we shall now describe the encoding of rules and facts in the extended system. Also, queries in the extended system can contain typed variables; we will discuss how the typed variables are interpreted and how such queries are answered.

5.1. Facts

Facts in the original system (Shastri & Ajjanagadde, 1990) are encoded by wiring up temporal pattern matchers, as shown in Figure 1(a). The extensions we have made to the system necessitate the addition of extra machinery in order to match correctly dynamic instantiations with long-term facts.

Concepts and instances can now accommodate k_1 instantiations. A fact should, therefore, be able to check if any one of these k_1 instantiations is in the required phase. This effect is realized by treating the output of concept clusters to be a bundle of k_1 links. Figure 10 shows the encoding of the fact $P(C_1, \ldots, C_n)$, where the outputs of $C_1, \ldots, C_n$ are considered to be bundles of $k_1 = 3$ wires. Further, given the capability of the system to encode multiple dynamic instances of a predicate, the dynamic instantiation which matches a long-term fact could occur in any one of the k_2 banks for that predicate. We therefore need a fact-pattern-matcher for each of the predicate banks. Thus, any fact $P(C_1, \ldots, C_n)$ will be encoded using k_2 τ-and nodes—one for each bank of P—as illustrated in Figure 10.

As mentioned in Section 3.1, a typed, universally quantified variable is treated as being equivalent to its type, and vice versa. Thus, the facts $\forall x{:}C_A, y{:}C_B\ P(x,y)$, $\forall x{:}C_A\ P(x,C_B)$ and $\forall y{:}C_B\ P(C_A,y)$ are equivalent to $P(C_A, C_B)$. A typed, existentially quantified variable is encoded by creating a unique subconcept of the associated type. Thus, $\exists x{:}C_A\ P(x,C_B)$ is encoded as $P(C'_A,C_B)$, where C'_A is a unique subconcept of C_A. This interpretation forces all existential variables to be outside the scope of the universal variables in the fact. Further, any unspecified role in a fact is treated as being existentially quantified. For example, $\forall x{:}\text{Cat}$ preys-on(x,y) would be interpreted as 'every cat preys on *some* bird'.

5.2. Queries

With the introduction of the type hierarchy, the extended system can answer queries with typed variables. Though the system can deal with both *yes–no* and WH queries, we shall concern ourselves only with *yes–no* queries.

Consider a query $P(\ldots, x, \ldots)$? where x is a typed variable of type C_A, filling the ith argument of P. To pose the query to the system, the enabler $e{:}P$ of predicate P is first activated. Depending on whether x is universally or existentially quantified, we proceed as follows:

- If x is universally quantified—i.e. the query is of the form $\forall x{:}C_A$ $P(\ldots, x, \ldots)$?—then C_{A1} and $C_{A1\uparrow}$ (i.e. the $C_\downarrow$ and $C_\uparrow$ nodes in the first bank of the entity cluster for C_A) are set to fire in synchrony with the ith argument of P. In order to verify if $P(\ldots, C_A, \ldots)$ is true, we need to check if $P(\ldots, C, \ldots)$ is asserted in the system where C is either C_A or an ancestor of C_A. Activating C_{A1} and $C_{A1\uparrow}$ achieves exactly this goal, by virtue of the activation control mechanism (Section 3.2).

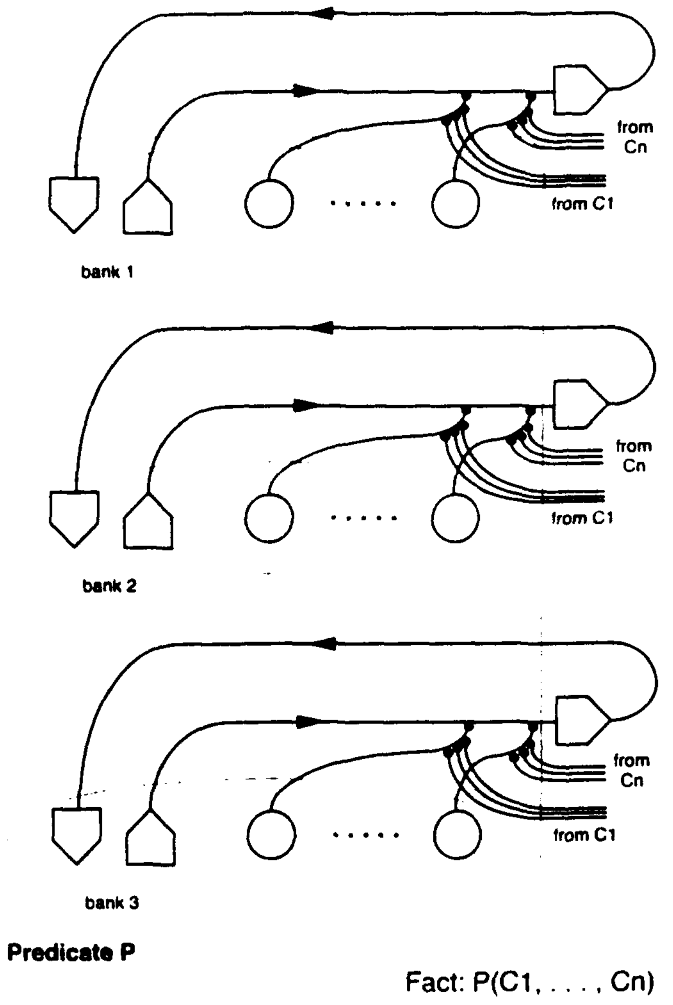


Figure 10. Encoding long term facts in the extended reasoning system. The fact encoded is $P(C_1, \ldots, C_n)$. We have assumed that $k_1 = k_2 = 3$.

- If x is existentially quantified—i.e. the query is of the form $\exists x{:}C_A\ P(\ldots, x, \ldots)?$—then C_{A1} and $C_{A1\downarrow}$ are set to fire in synchrony with the ith argument of P. This causes activation to spread to the ancestors, descendants and the ancestors of the descendants of C_A. As stated in Section 3.2, the propagating activation searches for facts which would render the current query true.

A more detailed justification of the correctness of the above procedure can be found in Mani and Shastri (1991).

Just as for facts, concepts directly specified in the query predicate are a shorthand for universal typed variables—i.e. $P(\ldots, C_A, \ldots)?$ is the same as $\forall x{:}C_A\ P(\ldots, x, \ldots)?$ Universally quantified variables are interpreted to be within the scope of the existentially quantified variables. Untyped variables are unspecified roles, and hence will not be assigned a phase when communicating the query to the network.

5.3. Rules

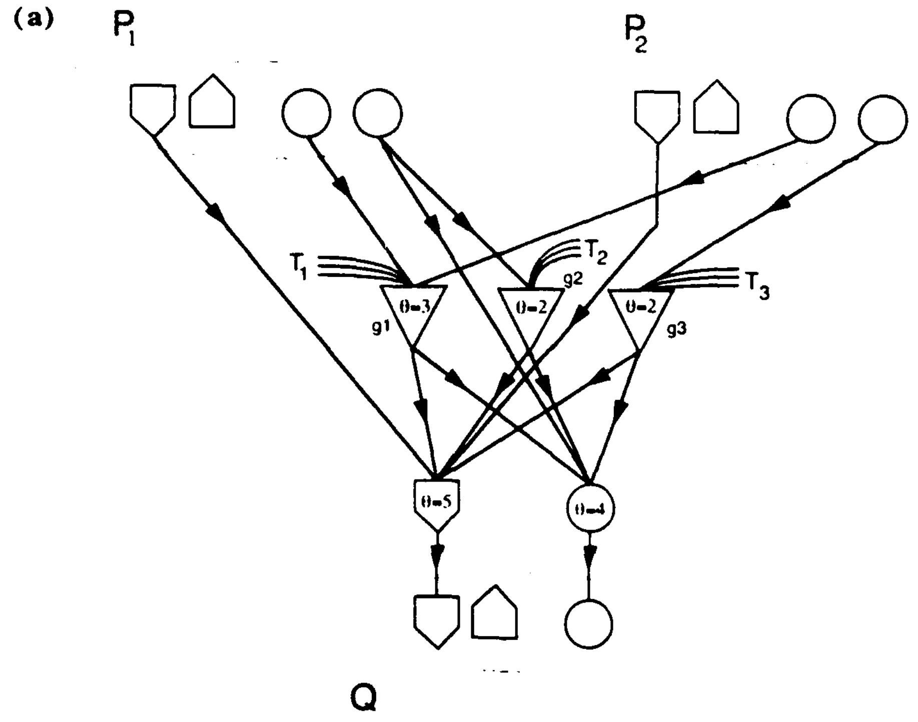
Rule encoding in the basic system is illustrated in Figure 1(a), and a detailed account can be found in Shastri and Ajjanagadde (1990). Here we shall consider the capabilities and complications introduced by the extended system.

The type hierarchy can be used to impose type restrictions on variables occurring in rules, for both forward and backward reasoning. To utilize this feature, we need to modify the encoding of rules. In a forward reasoning system, a rule is encoded by introducing a τ-or node to perform type checking for the argument in question. Figure 11(a) shows the encoding of the rule $\forall x{:}T_1, y{:}T_2\ \exists z{:}T_3\ P_1(x,y) \wedge P_2(x,z) \Rightarrow Q(y)$. Here, g_1, g_2 and g_3 are τ-or nodes for type checking which turn ON only if the corresponding predicate arguments are bound to objects of the right type. For example, g_3 would go ON only if the second argument of P_2 and at least one of the instantiations of T_3 are in synchrony—which is to say that the argument is bound to an object of type T_3. As indicated in the figure, links from T_1, T_2 and T_3 are bundles of k_1 wires each. It is also evident from Figure 11(a) that the rule will not fire—and predicate Q will not go active—unless all the g-nodes (g_1, g_2 and g_3) are active. In a backward reasoner, the strategy is similar, except for two basic differences. First, type checking for a typed universally quantified variable is enforced by a bundle of k_1 inhibitory links from the concept (see Figure 11(b)) representing the type of the concerned argument. Second, for a typed, existentially quantified variable, the inhibitory links for type enforcement are derived from a unique subconcept of the associated type.[12] The network which implements the rule $\forall x{:}T_1\ \exists y{:}T_2\ P(x) \Rightarrow Q(x,y)$ for backward reasoning is sketched in Figure 11(b). Any type mismatch causes g_1 to block further propagation of activation. Thus, in both the forward and backward reasoners, the encoding mechanism ensures that a rule fires only if all typed arguments are firing in synchrony with their respective types.

When multiple instantiation of predicates is introduced, the rule connectivity indicated above will need to be replicated k_2 times and rule wiring will need to be routed through the M-switch. Figure 8 illustrates rule encoding at a very gross level. Figure 12 gives a more detailed description of rule encoding in the extended system. Figure 12 depicts the encoding of the rule $\forall x,y\ P(x,y) \Rightarrow Q(y,x)$.

Each bank of predicate Q is connected to an input bank in every ensemble of the switch for P. Consider the connection from the ith bank of Q to the corresponding input bank in the jth ensemble of the switch for P. The input cable from bank i of Q connects to the input bank as though the input bank itself represented the predicate P. Thus, the connection pattern between the bank of Q and the input bank is identical to the connection pattern between the actual predicates in the system of Shastri and Ajjanagadde (1990). In particular, for the example in Figure 12, we have the following connections for $1 \le i \le k_2$ and $1 \le j \le k_2$:

- The enabler $e{:}Q_i$ of the ith bank of Q is connected to the enabler in the corresponding input bank of the jth ensemble of the switch for P.
- The collector in the same input bank is linked to $c{:}Q_i$, the collector of the ith bank of predicate Q.
- The first argument of the ith bank of Q connects to the second argument in the input bank while the second argument of the predicate bank connects to the first argument of the input bank.



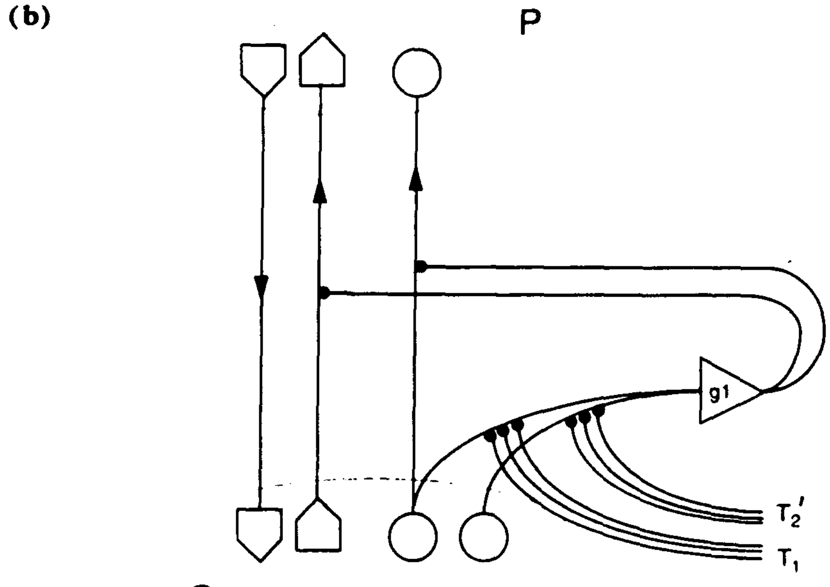


Figure 11. (a) Network encoding the rule $\forall x{:}T_1,\ y{:}T_2\ \exists z{:}T_3\ P_1(x,y) \wedge P_2(x,z) \Rightarrow Q(y)$ in a forward reasoning system. g_1, g_2 and g_3 are τ-or nodes which perform type checking, apart from enforcing other constraints; θ represents the node threshold. (b) Network encoding the rule $\forall x{:}T_1 \exists y{:}T_2\ P(x) \Rightarrow Q(x,y)$ in a backward reasoning system. T_2' is a unique subconcept of T_2; g_1 is a τ-or node. We have assumed that $k_1 = 3$.

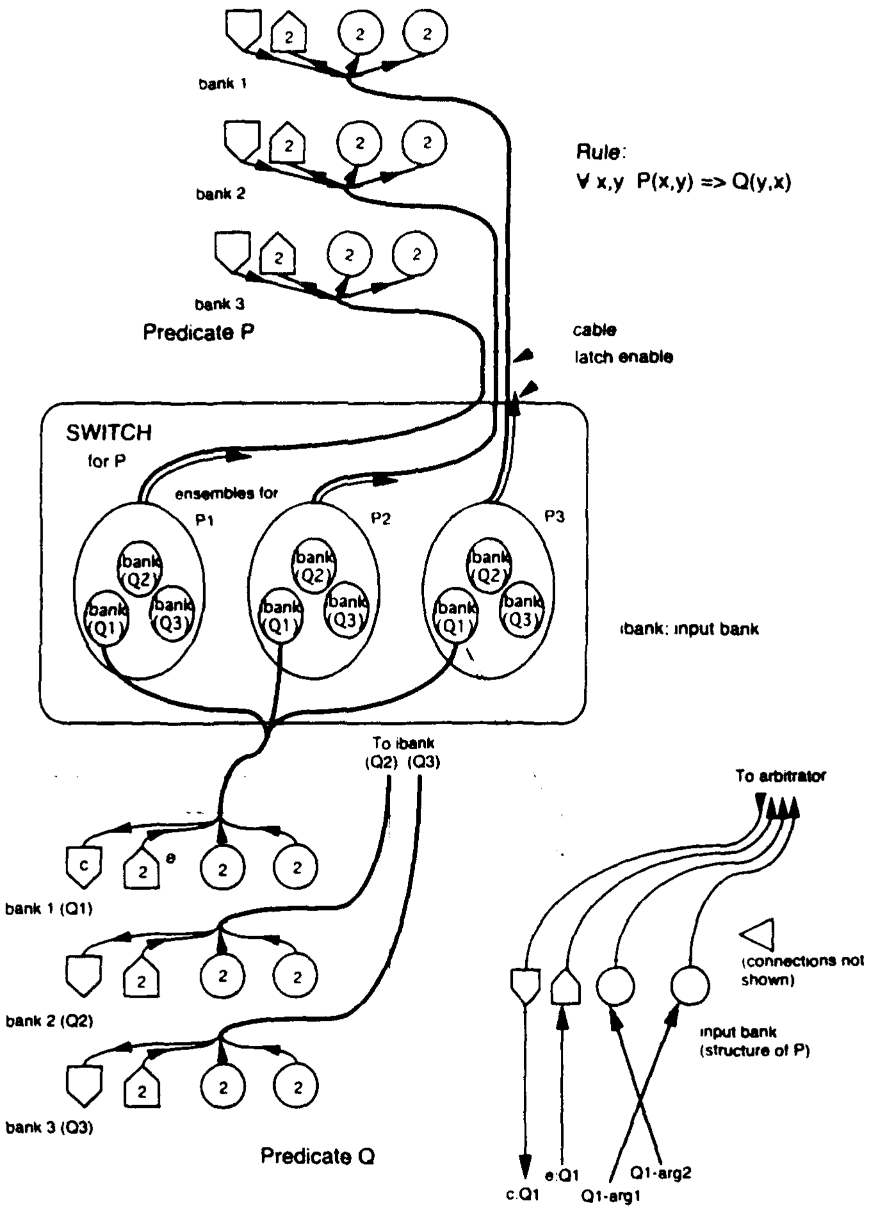


Figure 12. Encoding rules in the extended reasoner with multiple instantiation of predicates. The rule encoded is ∀x,y P(x,y) ⇒ Q(y,x). Nodes marked with a '2' have a threshold $\theta = 2$. To avoid cluttering, only part of the connections are indicated.

Since the *i*th bank of Q is connected to an input bank in every ensemble of the switch, the collector $c{:}Q_i$ of this bank receives inputs from the respective collectors in the input banks of all the ensembles of the switch. The τ-or unit associated with the input bank ensures that the collector $c{:}Q_i$ is activated if and only if the instantiation received by the input bank has been channeled to the predicate, and the collector in the corresponding bank of predicate P is active (refer to Figure 9(b)). In other words, the predicate collector $c{:}Q_i$ would be activated if the activation of Q_i has been channeled to P_i and the collector $c{:}P_i$ is active. The combination of collectors in the arbitrator and the input bank therefore serve as a mechanism for transmitting the state of the collector $c{:}P_i$ to the collector $c{:}Q_i$ of predicate Q.

Rules with multiple predicates in the antecedent are handled by an extension of the above procedure. Figure 13 gives a network which encodes the rule ∀x,y,z P(x,y) ∧ Q(y,z) ⇒ R(x,y,z). The $g3_i$ nodes[11] check that the dynamic activation of the ith bank of R follows from the facts for both P and Q. Note that $g3_i$

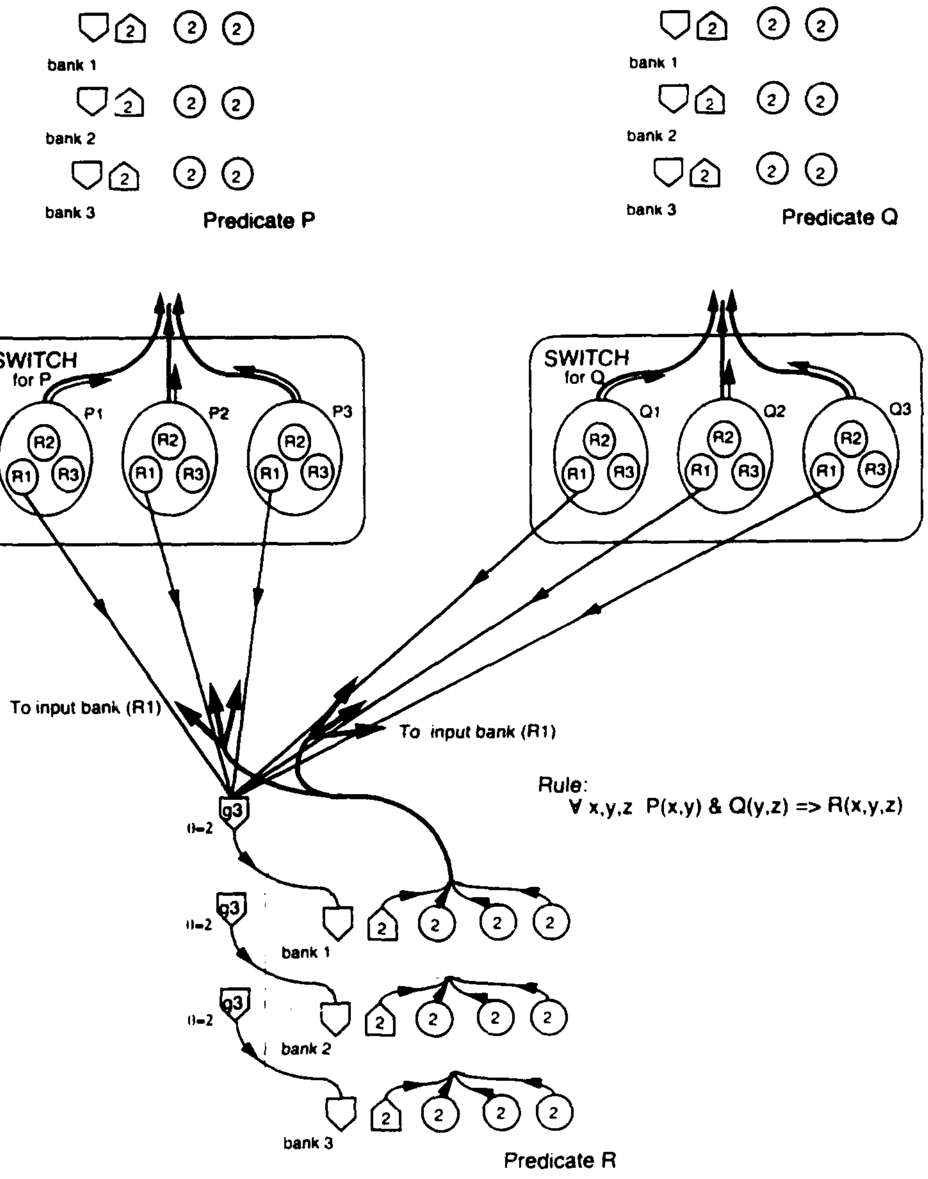


Figure 13. Encoding rules with multiple predicates in the antecedent. The rule encoded is ∀x,y,z P(x,y) ∧ Q(y,z) ⇒ R(x,y,z). The nodes marked g_3 are τ-and nodes; θ represents the node threshold. Nodes marked with a '2' have a threshold $\theta = 2$. To avoid cluttering, only relevant connections are indicated.

will become active irrespective of which banks of P and Q contain the activation which triggered the required facts.

The encoding of a rule with repeated variables in the consequent, existential variables in the consequent and constants[14] in the consequent is shown in Figure 14. The output of the $g1$ nodes inhibits the links from Q to the predicate(s) in the antecedent of the rule. Note that the scheme is identical to the one used to handle such conditions in Shastri and Ajjanagadde (1990), except that we repeat the scheme for each of the k_2 banks. Further, the inhibitory links from entities would be bundles of k_1 links.

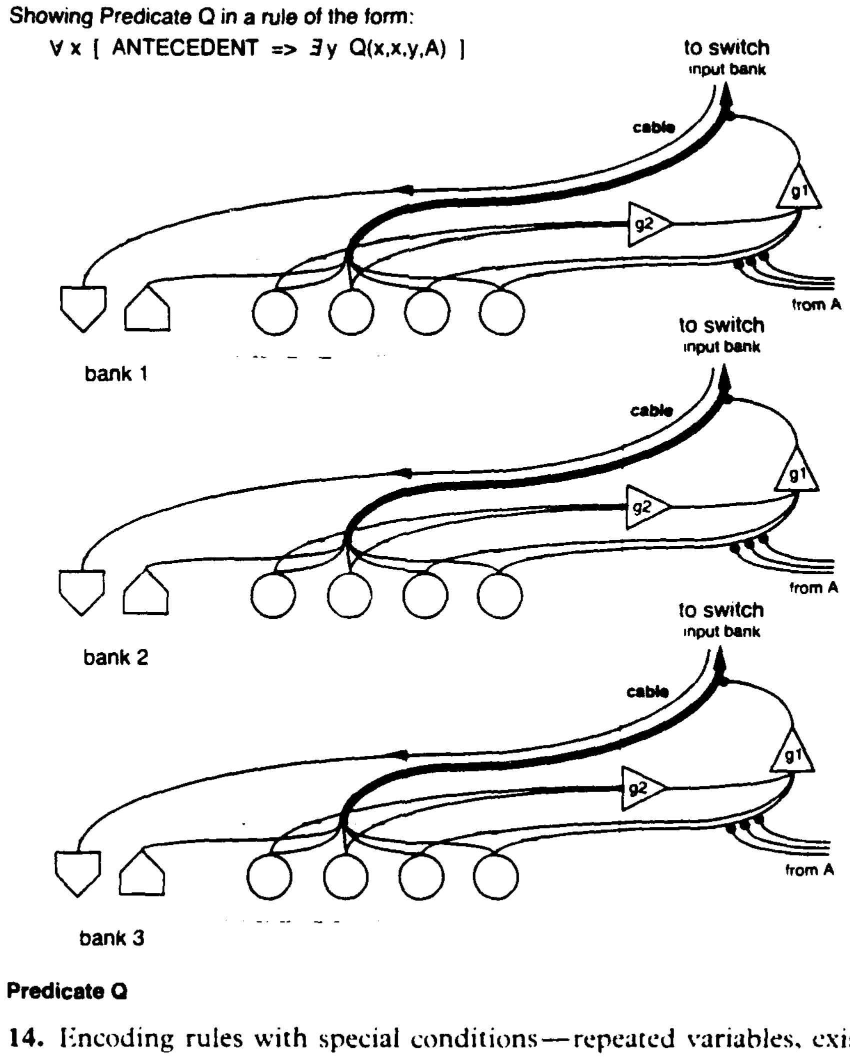


Figure 14. Encoding rules with special conditions—repeated variables, existentially quantified variables and constants in the consequent—in the backward reasoner. The rule encoded is $\forall x$ ANTECEDENT $\Rightarrow \exists y\ Q(x,x,y,A)$. The antecedent of the rule has been left unspecified since the mechanisms used to handle the special conditions are confined to modifying the activation from the consequent predicate. The g_1 nodes are τ-or nodes. The g_2 nodes are like τ-or nodes except that they become active if they receive input in more than one phase within a period of oscillation.

5.4. Complexity of the Network

The extended reasoning system requires $O(k_1 \cdot \mathcal{C} + k_2 \cdot \mathcal{F} + k_2^2 \cdot \mathcal{P})$ nodes, where $\mathcal{C}$ is the total number of entities in the system, $\mathcal{F}$ is the total number of long-term facts present in the reasoner, and $\mathcal{P}$ is the sum of the arities of all predicates in the rule base. k_1 and k_2 are the multiple instantiation constants for the type hierarchy and the rule base, respectively. As in Shastri and Ajjanagadde (1990), the network complexity is at most linear in the size of the knowledge base.

As for time complexity, the system can answer queries in time proportional to the length of the shortest derivation, irrespective of the number of rules and facts encoded in the system. Compared with the original system, the constant of proportionality is now slightly larger, since we also need to consider the time required for activation to propagate through the switches. Given a predicate P, the best case propagation time for activation passing through its M-switch is proportional to n, the arity of P; in the worst case, propagation time is proportional to $k_2 \cdot n$. If we assume that $n_{\max}$ is the maximum arity of any predicate in the reasoning system, then the constant of proportionality for the time complexity will be proportional to $n_{\max}$ (in the best case) or $k_2 \cdot n_{\max}$ in the worst case), irrespective of the predicate under consideration. The time taken for activation to traverse the type hierarchy is independent of k_1, and is only dependent on the number of is-a links that need to be traversed in order to answer the query. The time taken to answer a query will be proportional to the maximum of the time taken for activation to spread in (i) the rule base and (ii) in the type hierarchy.

5.5. Multiple Instantiation in a Forward Reasoning System

All along, we have looked at how the basic reasoning system could be extended to accommodate multiple instantiations of predicates in the backward reasoner. We now consider issues that arise when incorporating multiple instantiation of predicates in the forward reasoner.

In a forward reasoning system, predicates have the same structure as in the backward reasoning system. As before, every predicate has an associated multiple instantiation M-switch.[15] Rules with a single predicate in the antecedent can be encoded directly: each bank of the antecedent predicate is connected to input banks in every ensemble of the M-switch for the consequent predicate. Rules with multiple predicates in the antedecent, however, require special consideration. Suppose we have a rule of the form $\forall$x,y,z P(x,y) $\wedge$ Q(y,z) $\Rightarrow$ R(x,y,z). Suppose also that we are given the dynamic facts P(A,B) and Q(B,C). Then we should be able to conclude R(A,B,C). But the dynamic fact P(A,B) could be represented in any of the k_2 banks allocated for P. Similarly Q(B,C) could be active in any of the k_2 banks allocated for Q. To conclude R(A,B,C), we would need to pair each bank of P with all the banks of Q and check if the second argument of P is the same as the first argument of Q; in other words, we need to check if the second argument of P_i is the same as the first argument of Q_j for $1 \le i, j \le k_2$. The obvious solution to this problem requires $O(k_2^m)$ nodes and links to encode each multiple antecedent rule, where m is the number of predicates in the antecedent of the rule and k_2 is the multiple instantiation constant for the rule base. Typically, we expect the value of k_2 to be around 3, as argued in Shastri and Ajjanagadde, (1993a), and m to be around 2. Generally, in a rule containing an antecedent with several predicates, most of the antecedent predicates function to specify constraints on the arguments of one or two key predicates. Since the reasoning system can handle rules with typed variables, most of the predicates enforcing type constraints can be replaced by typed variables. For example, the rule ∀x,y collide(x,y) ∧ animate(x) ∧ solid-obj(y) ⇒ hurt(x) with three predicates in the antecedent is equivalent to the simple rule ∀x:animate, y:solid-obj collide(x,y) ⇒ hurt(x). The latter rule can be directly encoded in the extended reasoning system. Even if this 'compression' of the antecedent were not possible, we could always introduce dummy predicates and split a rule with several predicates in the antecedent into several rules with just a few predicates in the antecedent. Thus, with typical values of $k_2 \approx 3$ and $m \approx 2$, the extra cost of encoding rules in the forward reasoner with multiple instantiation is a factor of about 10 ($\approx 3^2$).

Special conditions in a rule (like repeated variables, existential variables in the antecedent, constants in the antecedent, etc.) can be handled as usual, before connecting a predicate bank to the input banks in the M-switch.

Incorporating multiple instantiation in the forward reasoner gives us the capability to encode rules like ∀x,y,z loves(x,y) ∧ loves(y,z) ⇒ jealous(x,z), and infer jealous(John,Tom) given loves(John,Mary) and loves(Mary,Tom).

5.5.1. Forward and backward reasoning. The structure of the M-switch used in both the forward and backward reasoners is identical. The differences in connectivity in the forward and backward reasoners arise from the differences in rule encoding. Further, handling special conditions in rules—like repeated variables, constants, existential variables, etc.—are dealt with differently in the forward and backward reasoners. Despite the similarities in the M-switch structure, the incompatible differences in the network structure of the forward and backward reasoners would make it difficult to use the same network for both forward and backward reasoning.

By carefully defining the interface between the forward and backward reasoners, we could have a system with both the forward and backward reasoners functioning independently and at the same time exchanging inferences and predicts, thereby complementing each other. Work is being done on developing such a system.

5.6. Constraints

The original system described in Shastri and Ajjanagadde (1990) and the extended system described here are tractable, limited inference systems (Shastri, 1993a). Though the system can handle a large class of rules and facts, there are some constraints on the form of rules and facts. A brief description of these constraints is provided here. A more detailed description along with psychological implications of these constraints can be found in Shastri and Ajjanagadde (1993a) and Shastri (1992).

In a backward reasoning system, where activation flows from the consequent predicate to the antecedent predicate(s), any predicate argument in the antecedent that requires some condition to be enforced must occur in the consequent and get bound during a given episode of reasoning. Thus, typed variables, repeated variables, existential variables and constants which occur in the antecedent of a rule must occur in the consequent and get bound during any episode of reasoning.

For example, the rule ∀x,y,z loves(x,y) ∧ loves(y,z) ⇒ jealous(x,z), cannot be encoded in the backward reasoner since the antecedent has a repeated variable *y* which does not occur in the consequent. In fact, it has recently been shown (Deitz *et al.*, 1993) that the repeated variable constraint—repeated variables in the antecedent must occur in the consequent and get bound during any episode of reasoning—is essential in order to draw inferences in time independent of the size of the knowledge base.

Constraints similar to the above hold for the forward reasoner: typed variables, repeated variables, existential variables and constants which occur in the consequent must occur in the antecedent and get bound during any episode of reasoning. Thus, the rule ∀x,y,z ∃t_1,t_2 move(x,y,z) ⇒ present(x,y,t
present(x,z,t_2) cannot be used for forward reasoning since the existential
ables t_1 and t_2 do not occur in the antecedent.

While representing facts and posing queries which involve typed variables, only those situations where all the universally quantified typed variables are within the scope of the existential typed variables can be represented. Further, concepts and predicates can only represent a limited number of dynamic instantiations.

The reasoning system also introduces the constraint that only a small number of entities can be simultaneously active at any given time. This may not be restrictive for any given episode of reasoning, but can be limiting when complex, interlinked chains of reasoning are required. In such cases, the set of entities in 'focus' must keep changing dynamically, recycling the available phases (see Section 5.7).

5.7. Significance of the Constraints

The reasoning system is psychologically and cognitively plausible in that it provides verifiable predictions and pointers which further our understanding of reflexive reasoning (Shastri, 1992; Shastri & Ajjanagadde, 1993a). The constraints listed in Section 5.6 predict what kinds of rules may participate in reflexive reasoning and what kind of rules will need 'reflective' reasoning. An example to wit would be the rule 'if *x* loves *y* and *y* loves *z*, then *x* is jealous of *z*'. Suppose we are told that 'John loves Mary' and 'Mary loves Tom' then we can reflexively infer that 'John is jealous of Tom'. But if we assume that the facts 'John loves Mary' and 'Mary loves Tom' are encoded as long term knowledge, we will find it difficult to answer the query 'Is John jealous of Tom?' in a reflexive manner. Observe that the rule 'if *x* loves *y* and *y* loves *z*, then *x* is jealous of *z*' can be used reflexively in the forward reasoner but not in the backward reasoner since the antecedent contains a repeated variable *y* which does not occur in the consequent (Section 5.6).

Another prediction introduced by the system is based on the multiple instantiation constraint. A predicate cannot be instantiated more than a (small) fixed number of times. Thus, we would expect to have difficulty dealing with too many instantiations at once. For example, we would normally find it difficult to answer questions about who loves whom reflexively after having been told (without repetitions) that: Susan loves Tom, John loves Lisa, Tom loves Mary and Clara loves Tom.

One of the constraints stated in Section 5.6 was that only a small number of distinct entities (seven to ten) can be simultaneously active in the system, at any given time. When taken together with other memory mechanisms, we argue that this is not as restrictive as it sounds. As discussed in Shastri and Ajjanagadde (1993a), when involved in tasks which require keeping track of a large number of entities over reasonable time spans—like reading a novel or participating in a conversation—those dynamic facts that are relevant could enter a medium-term memory where they would be available for a much longer time. Some of these facts might even be converted to long-term facts, which persist for a long time. There could arise situations—which require complex, interlinked chains of reasoning—where the total number of entities could exceed the limit imposed by our system. In such cases, the set of entities in 'focus' must keep changing dynamically, recycling the available phases. Identifying mechanisms that underlie such internal 'shifts of attention' and cause the system's activity to evolve smoothly remains a challenging open problem.

These constraints also have implications for other 'reflexive' processing phenomena besides reasoning. Henderson's (1993) work on parsing shows that the above constraints help in explaining some of the limitations of human parsing by modeling several linguistic phenomena involving long distance dependencies, garden path effects and our limited ability to deal with center-embedding.

5.8. Simulations

The reasoning system has been tested using a simulator (Mani, 1991) developed to run on the Rochester Connectionist Simulator (Goddard *et al.*, 1989). The simulator runs as a 'shell' on top of the Rochester simulator. It provides an input language for entering rules, facts and queries. A network encoding the input knowledge is automatically built. The simulator can construct stand-alone forward or backward reasoning systems, or a combined resoning system with the forward and backward reasoners forming independent layers. The simulation can be run interactively and the progress of the simulation monitored using graphic displays.

A knowledge base containing about 100 rules, 25 facts and 50 is-a facts has been simulated. The resulting network requires about 7100 nodes for the backward reasoner, about 8700 nodes for the forward reasoner and about 1200 nodes for encoding the type hierarchy. The node count for the backward reasoner includes nodes used to encode facts. The type hierarchy contains a total of about 60 concepts and instances. Based on these simulations, it can be argued that the reasoning system can draw a class of inferences in about a few hundred milliseconds. In a system made up of slow 'neurons', with a firing period π of about 20 ms and with the assumption that ρ-btu nodes can synchronize within two firing periods, we can arrive at the following timing estimates: the system takes about 260 ms to infer that 'John is Mary's spouse' given 'Mary is John's spouse'. Given that 'John bought a novel', the system takes about 320 ms to conclude that 'John owns a book'. With 'John bought a novel' encoded as a long-term fact, the system can answer 'yes' to the queries 'Did John buy a novel?' and 'Does John own a book?' in 140 ms and 420 ms, respectively.

Prototype implementations of the reasoning system have been developed on the Connection Machines CM-2 and CM-5, and initial results have been very encouraging. The prototype system can encode knowledge bases containing over a hundred thousand rules and facts, and can answer queries requiring an inference depth of up to ten in times ranging from a few milliseconds to a few hundred milliseconds on the CM-5.

6. Related Work

Though there have been several attempts to develop connectionist rule-based reasoning systems, very few of the approaches concern themselves with reflexive reasoning. A detailed discussion of the advantages and disadvantages of several other approches—both connectionist and otherwise—can be found in Shastri and Ajjanagadde (1993a). Here, we shall limit the discussion to type hierarchy implementations, multiple dynamic instantiation of predicates, and the interaction between the type hierarchy and the rule base.

Early work like Fahlman (1979) and Shastri (1988) considered efficient and massively parallel implementation of **is-a** hierarchies. But these systems did not deal with the explicit representation of rules involving *n*-ary predicates.

DCPS, a distributed connectionist production system introduced by Touretzky and Hinton (1988), was limited in that it could only deal with single variable rules, and like a classical production system, could only fire one rule at a time. TPPS, described in Dolan and Smolensky (1989), is a production system which uses tensor products to encode dynamic bindings. Like DCPS, TPPS is also serial at the knowledge level in that it can only fire rules serially. The restrictive nature of these systems—both in terms of expressive power and in terms of effective use of parallelism—renders them unsuitable for successfully modeling reflexive reasoning. DCPS and TPPS use a distributed encoding in that arguments and fillers are represented as patterns of activation over groups of nodes. These systems, therefore, inherit the advantages and disadvantages of distributed connectionist systems. The advantages of distributed representations stem from their ability to capture similarity. The seriality (at the knowledge level) imposed by representing roles and fillers as patterns of activity over a common pool of nodes, however, constitutes the major disadvantage of using distributed representations.

The key to capturing similarity in a distributed representation is the use of shared representation. As stated in Shastri and Ajjanagadde (1993b), the type hierarchy in the reasoning system also leads to such a sharing of representation by viewing the encoding of an entity as a distributed pattern over the collection of nodes that make up the type hierarchy. If one augments the representation of types with attribute values (Shastri & Feldman, 1986; Shastri, 1988), then the 'distributed' nature of the representation of each entity becomes even more apparent. For predicate arguments, the reasoning system uses a representation where each role is localized in the abstract representation.[16] It is this abstract localization of roles that enables the system to (i) overcome the inherent seriality of distributed representations and (ii) support knowledge-level parallelism.

CONPOSIT, a system introduced by Barnden and Srinivas (1991) uses relative position encoding and pattern-similarity association to solve the variable binding problem. Issues that we have considered in this paper, like encoding the type hierarchy, dealing with typed variables, etc. are not considered in Barnden and Srinivas's paper (1991). Rather than reflexive reasoning, CONPOSIT is tailored to handle a more complex class of rules and seems to be better suited for complex reflective reasoning—reasoning which requires reflection, conscious thought and deliberation. CONPOSIT is also serial at the knowledge level.

The connectionist reasoning system most similar in approach to the one proposed here is ROBIN (Lange & Dyer, 1989). ROBIN was developed to address ambiguity resolution in language understanding using evidential knowledge. In recent work, ROBIN has been extended to deal with case-based reasoning in combination with rule-based reasoning (Wharton *et al.*, 1992). Unlike the temporal synchrony approach, ROBIN can deal with an arbitrary number of entities during reasoning. ROBIN can do so because it permanently allocates a unique signature to each concept in the system, and sustains dynamic bindings by propagating these signatures. As discussed in Shastri and Ajjanagadde (1993a), there are certain drawbacks to propagating signatures in a connectionist network. Furthermore, the use of signatures does not lead to the type of psychological predicts than can be made using the temporal synchrony approach.

CONSYDERR is a connectionist rule-based reasoning system which uses a two-level architecture (Sun, 1991). One level uses a distributed representation while the other uses a localist representation. The two levels interact to provide a robust system that can handle partial, fuzzy and uncertain information. Though this system has an implicit type hierarchy built into the architecture, it does not consider how multiple dynamic instances of a predicate can be represented.

7. Conclusion

Adding a type hierarchy allows the reasoning system to represent is-a relationships efficiently and supports the occurrence of types as well as instances in rules, facts and queries. Being able to represent multiple dynamic instances of a predicate adds the capability to draw inferences using rules that capture symmetry, transitivity and recursion, provided the number of multiple instantiations required to draw a conclusion remains bounded. The extended reasoning system can therefore draw a much wider range of inferences. Though the resulting reasoner is relatively complex compared to the original system, the increased inferential power seems worth the added complexity, especially since we do not lose much in terms of efficiency.

The reasoner can perform high-level reasoning while still utilizing the massive parallelism inherent in connectionist systems. The resulting system can draw inferences in time which are independent of the size of the knowledge base. The system is also scalable in that very large knowledge bases can be handled tractably. Work is being done on mapping the system on to massively parallel SIMD and MIMD machines with the objective of attaining real-time performance (Mani, 1993).

Support for the neural plausibility of the system is provided by the fact that it is based on temporal synchrony. Recent neurophysiological (Gray *et al.*, 1991) data suggest that temporal synchrony may be used in the animal brain to represent dynamic bindings. A more detailed discussion of the cognitive, biological and psychological aspects of the reasoning system can be found in Shastri and Ajjanagadde (1993a).

This paper does not discuss the issue of learning. There has, however, been some preliminary work describing how such a system might convert dynamic bindings into medium-term facts (Geib, 1990). Developing effective learning algorithms is an exciting future research direction and an outline of a learning scheme for such a system is discussed in Shastri (1993b). Some of the other extensions being investigated include: dealing with facts and queries involving more flexible interleaving of quantifiers; expanding the system to allow property-value attachments to concepts in the type hierarchy; combining the forward and backward resoners to function as an integral reasoning system; and introducing evidential-preference rules, especially via learning.

Acknowledgements

This work was supported by NSF grant IRI 88-05465; ARO grants DAA29-84-9-0027 and DAAL03-89-C-0031; and ONR grant N00014-93-1-1149. This work has benefited from discussions and presentations at the Connectionist Group meetings at Penn. In particular, thanks to Tom Fontaine and Teow-Hin Ngair for their suggestions regarding the treatment of conflicts in the type hierarchy implementation. Thanks to Jeff Aaronson for discussions on an earlier version of the multiple instantiation switch. Thanks also to two anonymous reviewers for their thoughtful comments and suggestions.

Notes

1. There have been some notable exceptions; see Section 6.
2. Older-than(John's-father,John) follows from the knowledge that fathers are older than their children.
3. The solution to the multiple instantiation problem proposed in this paper is distinct from that outlined in Shastri and Ajjanagadde (1990), where two levels of temporal synchrony are used to deal with multiple instantiations of predicates.
4. In the rest of this paper, we assume that synchronous activity of nodes is oscillatory. As explained in Shastri and Ajjanagadde (1993b), however, oscillatory activity is not required for paper functioning of the model—the crucial requirement is that appropriate nodes synchronize. We have assumed oscillatory activity for convenience.
5. This is in keeping with the natural default meaning associated with statements like 'cats prey on birds', which generally means '*all* cats prey on *all* birds'.
6. This corresponds to the use of a skolem constant.
7. This applies to a predicate in the backward reasoning system. In a forward reasoner, the collector, *c:P*, and the arguments, $P_{arg_1}, \ldots, P_{arg_n}$, have a threshold $\theta = 2$. These nodes require a threshold of $\theta = 2$ in order to use a latch enable link to signal when the bank should go active (Figure 8).
8. Distinct from the T-switch used in the type hierarchy.
9. The enabler node *e:Arb* plays the role of the τ-or node for the first argument in the arbitrator. Thus, if we have a unary predicate, the latch enable link will originate from *e:Arb*.
10. If the node is receiving inputs in several phases, it picks one arbitrarily. In a simulation system, this could involve selecting a phase at random, selecting the first phase, selecting the last phase, and so on. The simulation system we use (Section 5.8) selects the first phase. In a physical system, however, the phase in which a node fires will depend on complex interactions between the relevant nodes and this interaction may even be chaotic.
11. Suppose the new instantiation arriving has already been assigned to bank j. In such a case, the inhibition on the corresponding input bank in ensemble j will be removed when the instantiation arrives. The input bank will become active and will blot out this instantiation from ensembles $j+1 \ldots k_{\nu}$, thereby automatically assigning this instantiation to bank j.
12. This is similar to the manner in which typed existential variables in a fact are interpreted.
13. $g3_i$ refers to the $g3$ node for the ith bank of the predicate.
14. Constants denote entities in the domain.
15. Though the multiple instantiation M-switch associated with a predicate in the forward reasoning system is structurally identical to the M-switch used in the backward reasoner, there are a few minor functional differences. Figure 9 shows connections in the context of a backward reasoning system. In a forward reasoner, the enabler in the arbitrator, *e:Arb*, connects to the collector of the associated predicate, while the enablers in the input banks receive inputs from the collectors of the input predicate banks. The collectors in the arbitrator and input banks are left unconnected, as are the enablers in the predicate banks.
16. Each role, however, can be represented by a cluster of nodes thereby providing a physically distributed representation.

References

Ajjanagadde, V. & Shastri, L. (1991) Rules and variables in neural nets. *Neural Computation*, **3**, 121–134.

Barnden, J.A. & Pollack, J.B. (1991) Introduction: problems for high-level connectionism. In J.A. Barnden & J.B. Pollack (Eds), *Advances in Connectionist and Neural Computation Theory, Volume 1.* Norwood, NJ, USA: Ablex Publishing Corporation.

Barnden, J.A. & Srinivas, K. (1991) Encoding techniques for complex information structures in connectionist systems. *Connection Science*, **3**, 269–315.

Deitz, P., Krizanc, D., Rajasekaran, S. & Shastri, L. (1993) A lower bound result for the common element problem and its implication for reflexive reasoning. Technical Report MS-CIS-93-73, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, USA.

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

Dyer, M.G. (1991) Symbolic neuroengineering for natural language processing: a multilevel research approach. In J.A. Barnden & J.B. Pollack (Eds), *Advances in Connectionist and Neural Computation Theory, Volume 1.* Norwood, NJ, USA: Ablex Publishing Corporation.

Fahlman, S.E. (1979) *NETL: a System for Representing and Using Real World Knowledge.* Cambridge, MA, USA: MIT Press.

Feldman, J.A. (1982) Dynamic connections in neural networks. *Bio-cybernetics*, **46**, 27–39.

Geib, C. (1990) A connectionist model of medium-term memory. Term Report, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, USA.

Goddard, N.H., Lynne, K.J., Mintz, T. & Bukys, L. (1989) Rochester connectionist simulator. Technical Report 233 (Revised), University of Rochester, Rochester, NY, USA.

Gray, C.M., Engel, A.K., Koenig, P. & Singer, W. (1991) Properties of synchronous oscillatory neuronal interactions in cat striate cortex. In H.G. Schuster & W. Singer (Eds), *Nonlinear Dynamics and Neural Networks.* Weinheim, Germany: VCH Publications.

Henderson, J. (1993) Description based parsing in a connectionist network. Forthcoming dissertation. Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, USA.

Lange, T.E. & Dyer, M.G. (1989) High-level inferencing in a connectionist network. *Connection Science*, **7**, 181–217.

Malsburg, C. van der (1986) Am I thinking assemblies? In G. Palm & A. Aertsen (Eds), *Brain Theory.* Berlin: Springer-Verlag.

Mani, D.R. (1991) Using the connectionist rule-based reasoning system simulator. Unpublished Report.

Mani, D.R. (1993) The design and implementation of massively parallel knowledge representation and reasoning systems: a connectionist approach. Dissertation Proposal, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, USA.

Mani, D.R. & Shastri, L. (1991) Combining a connectionist type hierarchy with a connectionist rule-based reasoner. Technical Report MS-CIS-91-33, University of Pennsylvania, Philadelphia, PA, USA.

Mani, D.R. & Shastri, L. (1992) Multiple instantiation of predicates in a connectionist rule-based reasoner. Technical Report MS-CIS-92-05, University of Pennsylvania, Philadelphia, PA, USA.

Shastri, L. (1988) *Semantic Networks: an evidential formulation and its connectionist realization.* San Mateo, CA, USA: Morgan Kaufmann.

Shastri, L. (1990) Connectionism and the computational effectiveness of reasoning. *Theoretical Linguistics*, **16**, 65–87.

Shastri, L. (1992) Neurally motivated constraints on the working memory capacity of a production system for rapid parallel processing: implications of a connectionist model based on temporal oscillations. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society.* Cognitive Science Society, Hillsdale, NJ: Lawrence Erlbaum.

Shastri, L. (1993a) A computational model of tractable reasoning—taking inspiration from cognition. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence.* San Mateo, CA, USA: Morgan Kaufmann.

Shastri, L. (1993b) Learning in SHRUTI. Technical report, International Computer Science Institute, Berkeley, CA, USA. Forthcoming.

Shastri, L. & Ajjanagadde, V. (1990) From simple associations to systematic reasoning: a connectionist representation of rules, variables and dynamic binding. Technical Report MS-CIS-90-05, University of Pennsylvania, Philadelphia, PA, USA.

Shastri, L. & Ajjanagadde, V. (1993a) From simple associations to systematic reasoning: a connectionist representation of rules, variables and dynamic bindings using temporal synchrony. *Behavioral and Brain Sciences*, **16**, 417–451.

Shastri, L. & Ajjanagadde, V. (1993b) A step toward modeling reflexive reasoning. Author response to Shastri and Ajjanagadde (1993a). *Behavioral and Brain Sciences*, **16**, 477–488.

Shastri, L. & Feldman, J.A. (1986) Semantic nets, neural nets and routines. In N. Sharkey (Ed.), *Advances in Cognitive Science, Vol. 1*. New York, NY, USA: John Wiley, pp. 158–203.

Sun, R. (1991) Connectionist models of rule-based reasoning. In *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*. Cognitive Science Society, Hillsdale, NJ: Lawrence Erlbaum.

Touretzky, D.S. & Hinton, G.E. (1988) A distributed connectionist production system. *Cognitive Science*, **12**, 423–466.

Wharton, C.M., Holyoak, K.J., Downing, P.E., Lange, T.E. & Wickens, T.D. (1992) Retrieval competition in memory for analogies. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*. Cognitive Science Society, Hillsdale, NJ: Lawrence Erlbaum.

Submitted to *Connection Science*

Recognizing Hand-printed Digit Strings Using Modular Spatio-temporal Connectionist Networks

Lokendra Shastri
International Computer Science Institute
1947 Center Street
Berkeley, CA 94704
(510) 642-4274
shastri@icsi.berkeley.edu

Thomas Fontaine
Tudor Investment Corporation
One Liberty Plaza
New York, New York 10006
thomas@tudor.com

Running Heading: HAND-PRINTED DIGIT RECOGNITION

Key Words: character recognition, digit recognition, word recognition, spatio-temporal neural networks, modular networks.

Abstract

We describe an alternate approach to visual recognition of hand-printed words, wherein an image is converted into a spatio-temporal signal by scanning it in one or more directions, and processed by a suitable connectionist network. The scheme offers several attractive features including shift-invariance and explication of local spatial geometry along the scan direction, a significant reduction in the number of free parameters, and the ability to process arbitrarily long images along the scan direction. Other salient features of the work include the use of a modular and structured approach for network construction and the integration of connectionist components with a procedural component to exploit the complementary strengths of both techniques. The system consists of two connectionist components and a procedural controller. One network concurrently makes recognition and segmentation hypotheses, and another performs refined recognition of segmented characters. The interaction between the networks is governed by the procedural controller. The system is tested on three tasks: isolated digit recognition, recognition of overlapping pairs of digits, and recognition of ZIP codes.

1 Introduction

A device capable of hand-print recognition has numerous applications in diverse areas such as postal sorting, print-to-voice transcription devices for the visually handicapped and human-machine interaction. [1] Given its importance and scope, the hand-print recognition problem has received considerable attention from researchers in the fields of pattern recognition and machine vision for over 30 years (e.g., (Bledsoe and Browning, 1959; Highleyman, 1961; Chow, 1962; Duda and Fossum, 1966; Munson, 1968; Pavlidis and Ali, 1975; Caskey and Jr, 1973; Yamamoto and Mori, 1979; Lam and Suen, 1988; Gader et al., 1991; Suen et al., 1992; Le Cun et al., 1990; Blackwell et al., 1992; Garris et al., 1992; Knerr et al., 1992; Fukushima et al., 1983; Burr, 1988; Denker et al., 1989; Shridhar and Badlerin, 1987; Fenrich and Krishnamoorthy, 1990; Keeler et al., 1991). In fact, it is perhaps one of the oldest and most explored problems in computer science. Yet the problem still remains largely unsolved. The difficulty in developing an effective solution to the problem can be attributed to the extremely high variance of unconstrained hand-print. This variance is due to a number of factors including: mechanical differences in stylus and writing surface, inter-author variations such as writing style, slant, and handedness, and even intra-author differences related to the purpose of writing and the mood of the author. Taken together, these factors introduce tremendous variability.

At the word recognition level the problem is further confounded due to variations in inter-character spacing. Since hand-print is not constrained to a uniform pitch, adjacent characters frequently touch or have overlapping bounding boxes. This gives rise to the character *segmentation* problem in which overlapping characters must be teased apart prior to recognition. Doing so, however is not so straightforward since overlapping characters lead to the segmentation and recognition dilemma: in order to segment a pair of characters, the characters must first be recognized, but in order to recognize the characters, they must first be segmented. Given this dilemma and the high degree of variance in hand-print, it is not surprising that the problem of hand-print recognition has remained largely unsolved.[2]

In this paper we investigate a particular approach to visual pattern recognition and describe its application to hand-printed character and word recognition. A key feature of our approach is that we treat spatial images as time-varying *spatio-temporal* signals and process them using appropriate connectionist networks. Some other salient features of our approach are (i) the use of a modular and structured approach for network construction and (ii) the integration of connectionist components with a procedural component to exploit the complementary strengths of both techniques.

[1]The scope of the problem can partially be gauged by the fact that the United States Postal Service alone processes over 80 million hand-printed pieces of mail every day.

[2]In contrast to hand-print recognition, excellent results have been obtained for reading of machine printed text where single character error rates as low as .01% have been reported (Schürmann, 1982).

Motivation

The variance inherent in pattern recognition problems such as hand-print recognition suggests the utilization of a system capable of learning complex, linearly non-separable, and fuzzy categories from examples. Connectionist networks offer a powerful framework for pursuing this approach and their strength has been demonstrated in a variety of pattern recognition problems including speech recognition (e.g., (Watrous, 1990; Waibel et al., 1989; Boulard and Morgan, 1994)), face recognition (e.g. (Cottrell and Metcalfe, 1991)) and even visual hand-print recognition (e.g., (Denker et al., 1989; Le Cun et al., 1990; Keeler et al., 1991)). Connectionist solutions are also attractive because once a network is trained, its simplicity, homogeneity, and parallelism can be exploited by VLSI technology. An entire network can be etched on a single microchip and, consequently, can attain very rapid recognition rates. Implementation is therefore relatively accessible, inexpensive, and attractive.

Visual pattern recognition schemes typically operate upon *static* images whereby an image is presented to a system as a time-invariant signal. This is also true of most connectionist approaches to hand-print recognition (e.g., (Denker et al., 1989; Le Cun et al., 1990; Keeler et al., 1991)). An alternate viewpoint is to consider an image to be a time-varying signal which is presented to a system in a piecewise fashion over time. For example, one could envisage a left-to-right scan of an image in which a system receives the ith column of the image at time i. Such a scan converts a static image into a *spatio-temporal* signal extending over several time steps. This approach offers several advantages: it leads to shift-invariance along the temporalized dimension, it explicates the local spatial relationships in the image along the temporalized dimension, it requires networks with fewer free parameters (weights), and it allows the assimilation of arbitrarily long images along the temporalized direction. These advantages are discussed in Section 2.

As is now widely recognized, training random or minimally organized networks using general purpose learning techniques is not a feasible methodology for obtaining scalable solutions to complex learning problems. We therefore adopt a more structured approach wherein we incorporate some prior structure in our networks and embed pretrained feature-detectors along with other "hidden" units. We also adopt a modular approach in order to make learning tractable. For example, instead of training a monolithic network for recognizing all the ten digits, we develop a separate network for each digit. Taken together, the use of structure and modularity allows the incorporation of domain knowledge, reduces the number of free parameters, and simplifies error analysis.

Although connectionist networks possess attractive features for pattern recognition applications, in many domains there is abundant domain knowledge that can be utilized effectively by traditional procedural techniques in a convenient manner.[3] Consider recognizing hand-printed ZIP codes, for example. A well-formed ZIP code will contain either five digits or nine digits (and perhaps a dash). This constraint can

[3]This does not mean that connectionist models cannot incorporate such knowledge. The issue is simply one of adopting a technique that is suitable for expressing and utilizing certain types of knowledge.

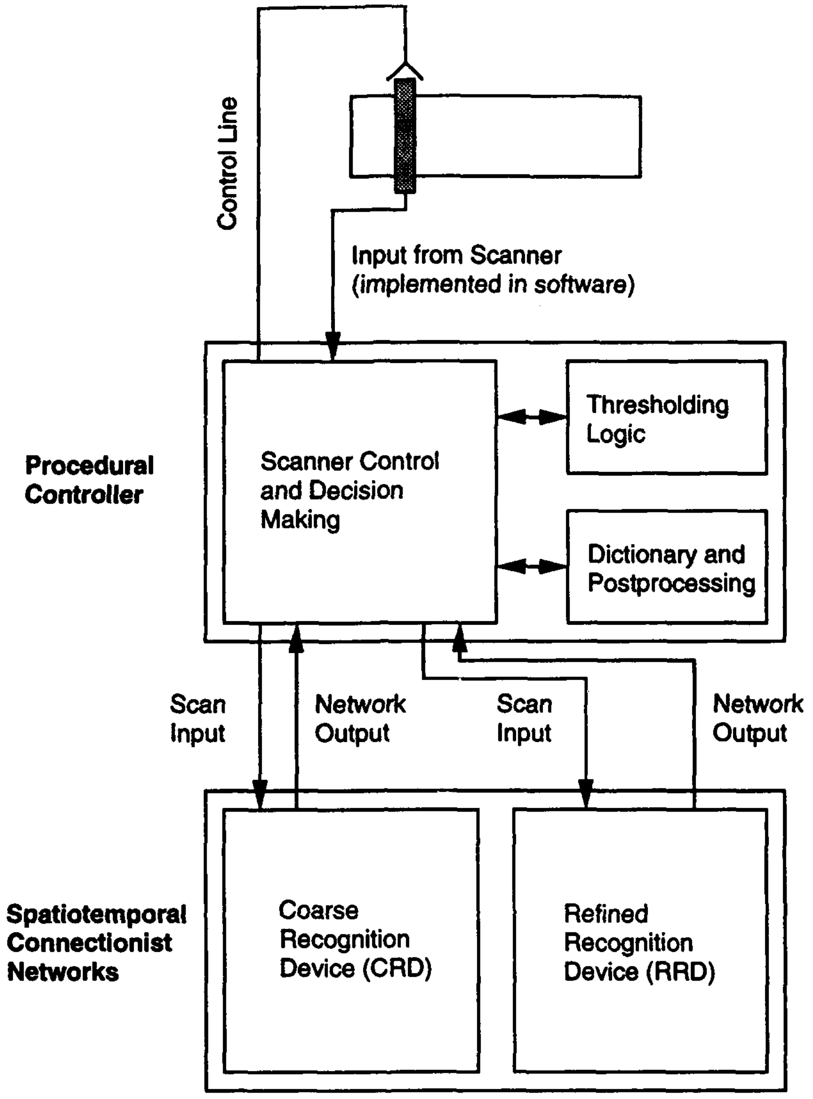

Control Line
Input from Scanner (implemented in software)
Procedural Controller
Scanner Control and Decision Making
Thresholding Logic
Dictionary and Postprocessing
Scan Input
Network Output
Scan Input
Network Output
Spatiotemporal Connectionist Networks
Coarse Recognition Device (CRD)
Refined Recognition Device (RRD)

Figure 1: An overview of the hybrid system.

easily be exploited by a procedural controller. Other domain specific knowledge (e.g., statistics gathered from envelopes arriving at particular postal branches in the case of ZIP codes) and standard dictionary based algorithms can also be implemented effectively using a procedural approach. This suggests a hybrid approach, wherein fast and robust connectionist networks perform recognition in concert with a procedural component capable of incorporating systematic domain knowledge, heuristics, and well-studied algorithms.

1.1 Preview

We have developed a system for hand-printed word recognition using the concepts described above. The system recognizes well-printed word images containing white space between characters as well as more difficult images in which characters are ill-formed, disjoint, or overlapping.

The system consists of two connectionist networks and a procedural controller (see Figure 1). One network, called the Coarse Recognition Device (CRD), assimilates a word image in a left-to-right fashion over time and performs coarse character recognition. While doing so, it also hypothesizes segmentation boundaries between characters. The other network, called the Refined Recognition Device (RRD), specializes in isolated character recognition, and attempts to classify portions of the image hypothesized to be characters by the CRD. The two networks are governed by a conventional procedural controller, capable of fusing signals emanating from the two networks while incorporating domain knowledge. The final recognition is the result of the combined effort of the three components. Our focus in this work has primarily been the development of the two connectionist components and the evaluation of the spatio-temporal approach since we perceived these to be the most challenging aspects our approach. Consequently, the procedural component has received only limited attention.

The system (without any high-level domain knowledge encoded in the procedural controller) was tested on three tasks: isolated digit recognition, recognition of overlapping pairs of digits, and recognition of ZIP codes. On a test set of 2,700 isolated digits, provided by the United States Postal Service, the system achieved a 96.0% accuracy. On a test set of 207,000 isolated digits, provided by the National Institute of Standards and Technology, a 96.5% accuracy was attained. Six sets of 500 images of digit pairs whose rectangular bounding boxes overlapped were synthesized from isolated digits for testing. The sets differed depending on the degree of overlap in their bounding boxes (0%, 5%, or 10% of the first box width). System accuracy ranged from 87.6% to 65.6%, and it was seen that performance on pairs drawn from the test set closely tracked performance on pairs drawn from the training set. Finally, recognition performance was measured on a set of 540 real-world ZIP code images, provided by the United States Postal Service. Using a criterion in which a ZIP code classification was deemed correct if and only if the produced digit string matched the *complete* ZIP code exactly, the system achieved a 66.0% accuracy. Note that the 66% rate is a "worst-case" measurement—it considers a classification of an entire ZIP code incorrect in the event that any constituent digit is incorrect.

The rest of the paper is organized as follows. In section 2 we present the spatio-temporal approach to pattern recognition and argue that it offers a number of advantages. In Section 3 we describe a hybrid and modular spatio-temporal system for hand-print recognition that instantiates this approach. We discuss the methodology for training and testing the system in Section 4 and present empirical results in Section 5. We conclude with a general discussion and an outline of future directions in Section 6.

2 The spatio-temporal approach

Visual pattern recognition schemes, including connectionist ones, typically operate on *static* images whereby an image is presented to the system as a time-invariant signal (Denker et al., 1989; Le Cun et al., 1990;

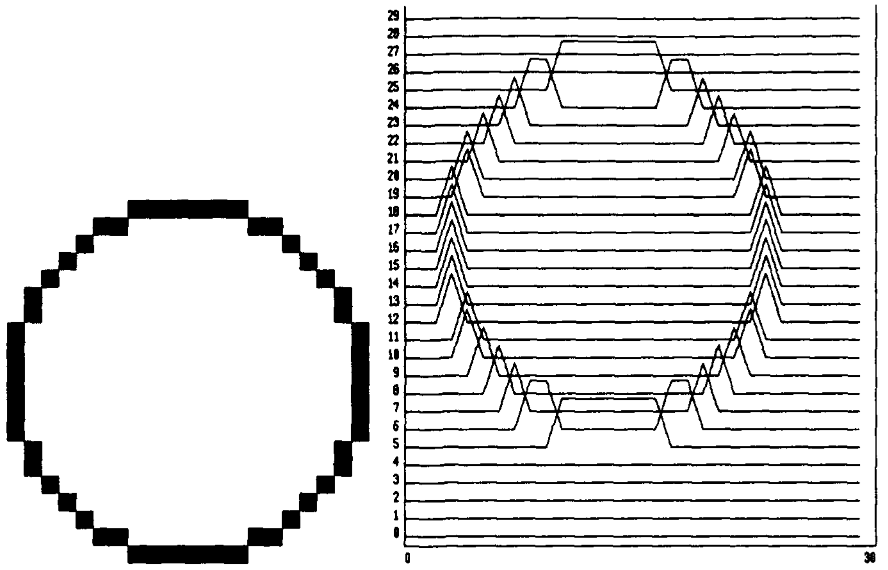


Figure 2: A static "0" image (left) and the spatio-temporal input generated by a left to right scan (right). In the latter, the vertical axis enumerates input units, the horizontal axis is time, and the third axis indicates the level of activation.

Martin and Pittman, 1990). This approach has produced good results in isolated character recognition and has also been applied with limited success to word recognition (Keeler et al., 1991).

An alternate approach is to convert an image into a time-varying signal by scanning it in one or more directions and presenting the resulting *spatio-temporal* signal to the recognition system. For example, if a system scans an $n * m$ image from left to right, it receives the n pixels in column i of the image at time i. This converts the static image into a spatio-temporal signal that extends over m time steps and has a spatial span of n. Figure 2 graphically illustrates this by showing the spatio-temporal signal generated by a left to right scan of a "0". The image of a "0" is shown on the left and the image as it would be received by a network's input units is shown to the right. The horizontal (x) axis represents time, while the vertical (y) axis enumerates 30 input units. The plot for each input unit depicts its activation level over time (the levels of activation can be viewed as being represented along the z axis orthogonal to the page).

2.1 Advantages of the spatio-temporal approach

Time-varying signals arise naturally in problems such as speech recognition and time series prediction where the input signal has an explicit temporal aspect. But what is their significance for visual recognition? We discuss the answer below and point out what we think are inherent advantages in considering images as spatio-temporal signals.

Most work on visual pattern recognition treats an image as a static two-dimensional pattern. Therefore the suggestion that images be treated as spatio-temporal signals may seem counter-intuitive. A little reflection, however makes it apparent that a static view of visual processing is unrealistic. In general, an agent must scan its environment in order to locate and identify objects of interest. Even in a more restricted setting such as recognizing ZIP codes on pieces of mail, a device must scan the face of the envelope to locate the region containing the ZIP code. Finally, even if the (starting) location is known, scanning is required if the image contains a number of objects. Observe that reading text essentially involves processing a *continuous* stream of visual data having an *arbitrary* extent. Thus scanning is an integral part of visual processing. On-line character recognition systems that use values of the position (and optionally, velocity and acceleration) of the pen over time to recognize characters (e.g., (Guyon et al., 1991; Schenkel et al., 1993)) can also be viewed as systems that "scan" the characters as they are being constructed over time.

Shift-Invariance

A recognition system which responds identically to an object regardless of the spatial location of the object, is *shift-invariant*. In pixel-level image recognition using traditional connectionist networks, the number and arrangement of input units typically correspond to the number and arrangement of pixels in the input image. Since an object may appear at different spatial locations in different images, the relevant data may be assimilated by different sets of input units. Hence, a method must be devised for recognition regardless of which set of input units receives the data. An obvious but significant advantage of our approach is that it naturally leads to a recognition system that is *shift invariant* along the temporalized axis(es). When an image is scanned, any 'white space' in the image generates a zero input and leaves the network state unaffected. Thus the network ignores 'white space' and responds to the object it is trained to recognize wherever (or whenever) it encounters that object in the image. Thus shift-invariance along the temporalized axis falls out as a natural byproduct of the approach.

The spatio-temporal approach explicates the image geometry

The local spatial relationships in the image along the temporalized dimension are naturally expressed in the scanned input. Consider a unit in the first hidden layer of a traditional (static) network. The activation received by this unit from units in the input layer are unlabeled levels of activation, and hence, this unit cannot determine which inputs come from spatially neighboring pixels. As far as the hidden unit is considered, the input it receives from an image I is indistinguishable from the input it would receive from the image I' obtained by permuting the pixels of I. Now consider a hidden unit in the spatio-temporal network. The inputs to this unit from two adjacent pixels (along the temporalized dimension) become available in adjacent time steps. Hence the spatio-temporal approach makes spatial locality explicit by mapping it into temporal locality.

Reduction in network complexity

In the spatio-temporal approach a spatial dimension is replaced by the temporal dimension and this leads to models that are architecturally less complex than similar models that use two spatial dimensions. This reduction of complexity occurs because the spatial extent of any feature in an object's image is much less than the spatial extent of the object's image. Consider the case where an object's image is $n * m$. Let the extent of the image along the temporalized dimension be m and let the maximum width of any feature along this dimension be k. Typically k will be significantly less than m. A traditional network for recognizing this object would require $n * m$ input units. Observe however, that the processing ability of a traditional network can be replicated by a spatio-temporal network containing only n input units connected to hidden units via a bundle of k links with propagation delays ranging from 1 to k.[4] The use of multiple links with varying delays allows a hidden node in the spatio-temporal network to receive inputs arising from a limited window (or receptive field) of height n and width k. The limited width of this receptive field, however, is sufficient since it exceeds the size of all features in the image! Furthermore, as scanning progresses, this receptive field slides along the scan direction and fully traverses the image. If we assume that hidden units act as feature detectors then the moving receptive field of a hidden node in the spatio-temporal model leads to an effective tessellation of the feature detector over the image *without* the actual (physical) replication of the feature detector.

In view of the equivalent processing power of a spatio-temporal network and a traditional network, it can be argued that while the number of links required by a spatio-temporal network is proportional to $n * k$, the number of links in a a traditional network will be proportional to $n * m$. Typically, k is much less than m and therefore the spatio-temporal model will require significantly fewer links. During training, the number of links in the network corresponds to the number of free parameters in a non-linear optimization process, and a substantial reduction in the number of free parameters can yield faster optimization.

Processing arbitrarily long inputs

A common difficulty of the connectionist approach to pattern recognition is that a network must have a fixed number of inputs, and thus must process images of a fixed size. This makes it difficult for a conventional connectionist model to recognize words — although progress has been made by replicating and tessellating network substructures to accommodate images with multiple characters (Keeler et al., 1991). In contrast, the ability to process arbitrarily long images is inherent in our approach, and offers an alternate means to process word images within a connectionist framework and relaxes the restriction of fixed-size inputs (see 3).

[4]The discussion assumes that input units are fully connected to hidden units. The basic point however, also holds for limited connectivity between input and hidden layers.

Addressing the segmentation and recognition dilemma

The use of scanning also partially solves the segmentation/recognition dilemma. Most vision systems perform a segmentation step and then attempt to recognize the segments. This approach is feasible as long as objects are non-occluding. If an image contains several objects that touch and/or overlap, segmentation becomes problematic and the system is faced with the segmentation/recognition dilemma. As explained in Section 3, the recognizer can continually update the activation level of its output units, as the image is being scanned from left to right and this activation trace may also be used to estimate the segmentation point.

2.2 Spatio-temporal networks

Processing a spatio-temporal signal requires a model capable of processing time-varying signals. A number of researchers have proposed network models to represent and process such signals (e.g., (Elman, 1990; Jordon, 1987; Lapedes and Farber, 1987; Mozer, 1989; Waibel et al., 1989; Watrous and Shastri, 1986). The connectionist model we employed was inspired by the *Temporal Flow Model* (TFM) which has achieved good results in speech recognition (Watrous, 1990; Watrous, 1991). TFM supports arbitrary link connectivity across layers, admits feedforward as well as recurrent links, and allows variable propagation delays to be associated with links. These features provide a means for smoothing and differentiating signals, measuring the duration of features, and detecting their onset. They also allow the system to maintain context over a window of time and thereby carry out spatio-temporal feature detection and pattern matching. Taken together, the use of recurrent links and variable propagation delays provide a rich mechanism for short-term memory, integration and context sensitivity — properties that are essential for processing time varying signals — and provides a potentially powerful mechanism for performing feature detection and pattern recognition.

Spatio-temporal networks also have a sound basis in biology. It is well known that circuits for auditory processing in animals make explicit use of propagation delays (e.g., see Edelman et al. 1988). Similarly, propagation delays, delay tuned neurons, and coincidence detectors are used by bats for echo-location and by the barn owl for localization of objects via the detection of differences in inter-aural timing (e.g., see (Carr and Konishi, 1990)).

3 The word recognition system

3.1 Overview

The complete system (refer to Figure 1) consists of three components: the Refined Recognition Device (RRD), Coarse Recognition Device (CRD), and Procedural Controller (PC). The system's ability to deal with disjoint as well as overlapping digits stems from the interaction between these components.

Without loss of generality, assume that an image is being scanned in one direction. The spatio-temporal

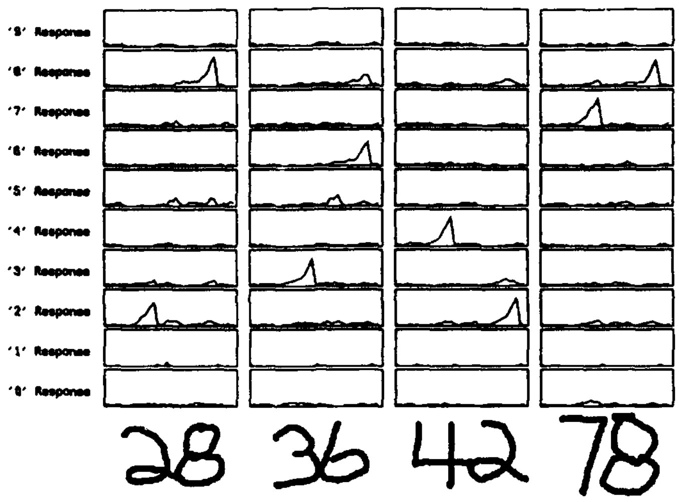


Figure 3: Output unit response of the Coarse Recognition Device in response to a set of images depicting touching or overlapping pairs of digits. Sharp peaks in response correspond to recognition of a digit and subsequent resetting of the CRD.

signal resulting from the scan is input to a CRD which is a spatio-temporal network trained to act as a coarse recognizer. The CRD has one output unit for each class in the domain. As the image is scanned, the activation level of each CRD output unit indicates the degree of support for the presence of a token of the associated class in the region currently being scanned. When the support for any class reaches a threshold, the scanning stops and the CRD *hypothesizes* the presence of a token of the appropriate class. At this time, the relevant region of the image is extracted and processed by the RRD, the refined recognition network which specializes in recognizing isolated digits. RRDs are also spatio-temporal networks which process an *extracted* region by scanning it in one or more directions. On the completion of processing, the RRD either confirms or rejects CRD's hypothesis. If the hypothesis is confirmed by the RRD, the system announces the presence of the appropriate digit at the appropriate location in the image and CRD continues its scan of the image. If the RRD rejects the hypothesis, it considers (overlapping) regions in the immediate vicinity of the region under consideration and tries to locate the hypothesized object. If the hypothesized object is still not found, CRD continues its scan of the image.[5]

The interaction between the CRD and RRD is mediated by the procedural controller (PC). It is the PC which detects that one of the CRD output units has reached threshold, extracts the relevant portion of the image, and passes it on the RRD.

[5]In the actual system implementation (see Section 5.3), the process described above is preceded by a *connected component extraction* step. A connected component is simply a set of "on" points in the image such that any two points belonging to the same component are connected by a path of adjacent "on" bits. Connected components can be extracted by a simple scan of the image and a parallel connectionist implementation is described in (Fontaine, 1993). Each connected component so obtained is first processed by the RRD. If the RRD recognizes a component as a digit with high confidence, the component is deemed to be that digit. All the remaining components are processed by the CRD and RRD in the manner described above.

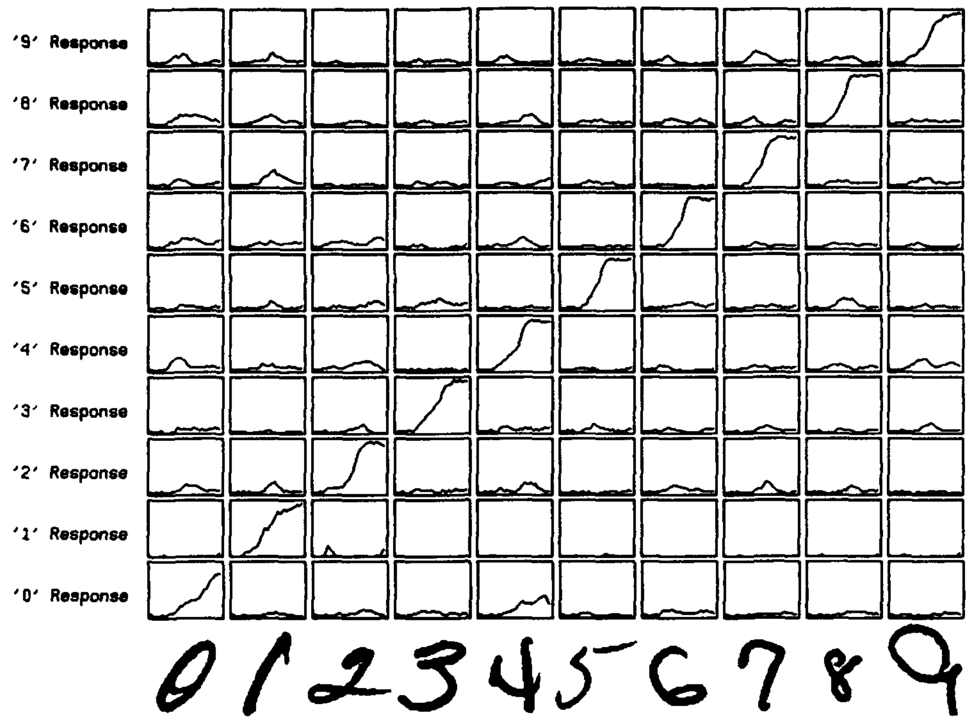


Figure 4: RRD output unit response to a typical set of ZIP code digit images

Figure 3 shows the response of the CRD in response to a set of touching and overlapping pairs of digits. Sharp peaks correspond to recognition of a digit and the subsequent resetting of the CRD by the PC. Figure 4 shows the output unit response of the RRD network to some typical isolated ZIP code digit images.

Basic architecture of CRD and RRD

Both CRD and RRD networks are spatio-temporal networks with multiple hidden layers, feedforward as well as recurrent connections, and multiple links – with variable delays – between units. Each network typically consists of four layers: an input layer, two hidden layers, and an output layer.

The number of units in the input layer is determined by the number of image pixels "seen" at each step of the scanning process. For example, if an $n * m$ image (i.e., an image with n rows and m columns) is scanned from left to right, the number of input units is n. If the image is scanned in multiple directions, there are separate banks of input units – one for each scan direction.

The first hidden layer is best viewed as a layer of feature detectors. Each unit in this layer has an associated *receptive field* and is expected to detect the occurrence of some salient feature(s) in this field. As pointed out in Section 2.1, the receptive field of a unit is temporalized and *moves* in the direction of scan during processing.

Most of the units in the feature detector layer are adaptable and during training, 'learn' to detect appropriate feature(s) in the image. In addition to these adaptable units, some pre-trained feature detectors can be embedded in the hidden layer. We do this by including units connected to input units via appropriately weighted links that enable these units to detect features such as oriented bars. The second hidden layer receives inputs from the feature detector units in the first hidden layer. Units in the second layer *integrate* the response of feature detectors and adapt so as to detect complex features and non-local feature combinations required to recognize objects in the image. We now describe each component in more detail.

3.2 Refined recognition device (RRD)

The RRD is responsible for accurate recognition of isolated hand-printed digits. We have developed the RRD in a modular manner in order to incorporate domain knowledge, reduce the number of free parameters, and simplify network analysis. The RRD consists of ten individually trained Single Digit Recognition Networks, each of which is responsible for the detection of a particular digit. Each Single Digit Recognition Network consists of four Single Scan Networks, each of which assimilates data from a different "scan" of the image. A Single Scan Network is constructed from a number of adaptable layers, operating in conjunction with a number of pretrained Feature Detection Modules. A Feature Detection Module is formed by the replication and tessellation of a pretrained Local Receptive Field.

Feature detection modules

Most numerals can be approximately written using four simple stylus strokes: horizontal, vertical, slash, and backslash. The simplicity and recurrence of these strokes suggests the utility of developing pretrained feature detection modules, which can be integrated into a larger network. A separate *Local Receptive Field* module (or LRF) was pretrained to detect each of these four features over a localized area.

The generic LRF module is seen in Figure 5. It receives input over a spatial field of 4 inputs, a temporal field of 4 time steps, and consists of 4 input units, 4 hidden units, and a single output unit. Hidden unit n receives information from all input units, and utilizes n links from each input unit, with respective delays of $1, 2, \ldots, n$, creating a spatial window of width n into the temporal signal. As long as a feature to be detected by an LRF is present in its 4 by 4 receptive field, the LRF will emanate an output signal, albeit with a slight lag. Various LRF modules for detecting horizontal, vertical, slash, and backslash strokes were trained using the same generic architecture.

Local detectors can be replicated to tessellate an entire "column" of the image. But note that the tessalation along the other dimension occurs implicitly when the image is scanned. We refer to a group of identical and tessellated LRFs as a *Feature Detection Module*, or FDM. An example of an FDM using 3 LRFs, with an input unit overlap of 2 and covering a receptive field of 8 inputs, is seen in Figure 6. The dashed box

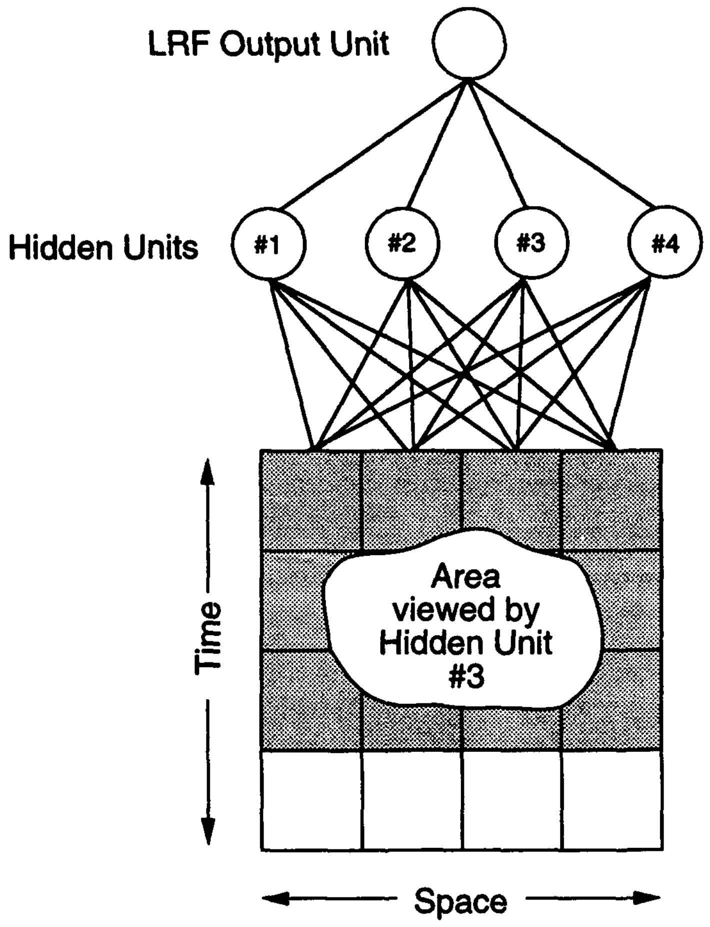


Figure 5: A generic Local Receptive Field (LRF)

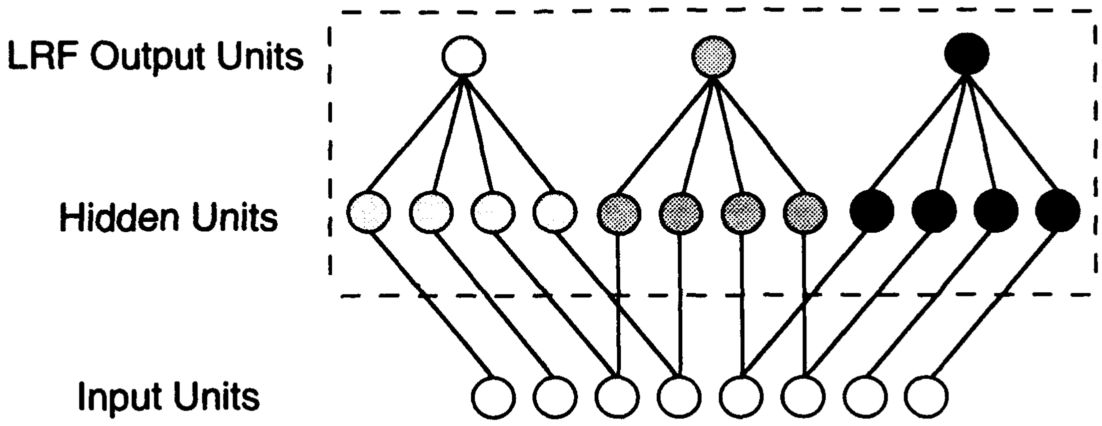


Figure 6: A generic Feature Detection Module (FDM)

demarcates the entire FDM.

A desirable trait of the feature detectors is their modularity. Each feature detector is composed from an LRF building block in a simple manner, and the number of useful feature detectors is limited only by the number of useful LRFs which can be developed. At a different level of modularity, the feature detection modules can be inserted into a larger network design. During optimization, the FDMs are masked out and are not considered part of the optimization (although they could be fine-tuned via training, if desired). This allows the incorporation of robust feature detectors which yield useful information without increasing the dimensionality of the optimization.

Single Scan Networks

The signal from each scan is processed by what we refer to as a *Single Scan Network* (SSN). Figure 7 illustrates the configuration of a SSN, referred to as a *Single Scan Network.* In this instance, the SSN operates on 20x20 images, using two pretrained FDMs (a horizontal and slash stroke detector), and several unstructured hidden layers. The input units pass information along links which are either frozen, if they are part of a pretrained FDM (dashed lines), or trainable, if they are "regular" links (solid lines). A local hierarchical structure is used to detect higher order features as information propagates towards the output unit.

A specific SSN used had the following architecture: Each LRF was connected to 4 (out of 20) adjacent input units. Each input unit to LRF connection consisted of 3 links having delays of 1, 3, and 5 respectively. Adjacent LRFs had an overlap of two input units. There were four Feature Detection Modules in the first hidden layer, with each FDM containing 9 LRFs. The second hidden layer was arranged in 4 banks of 6 units, with each bank receiving input from a corresponding FDM. Each unit in a bank received information from 4 contiguous FDMs via unit delay links. The units in the second layer of the Single Scan Networks

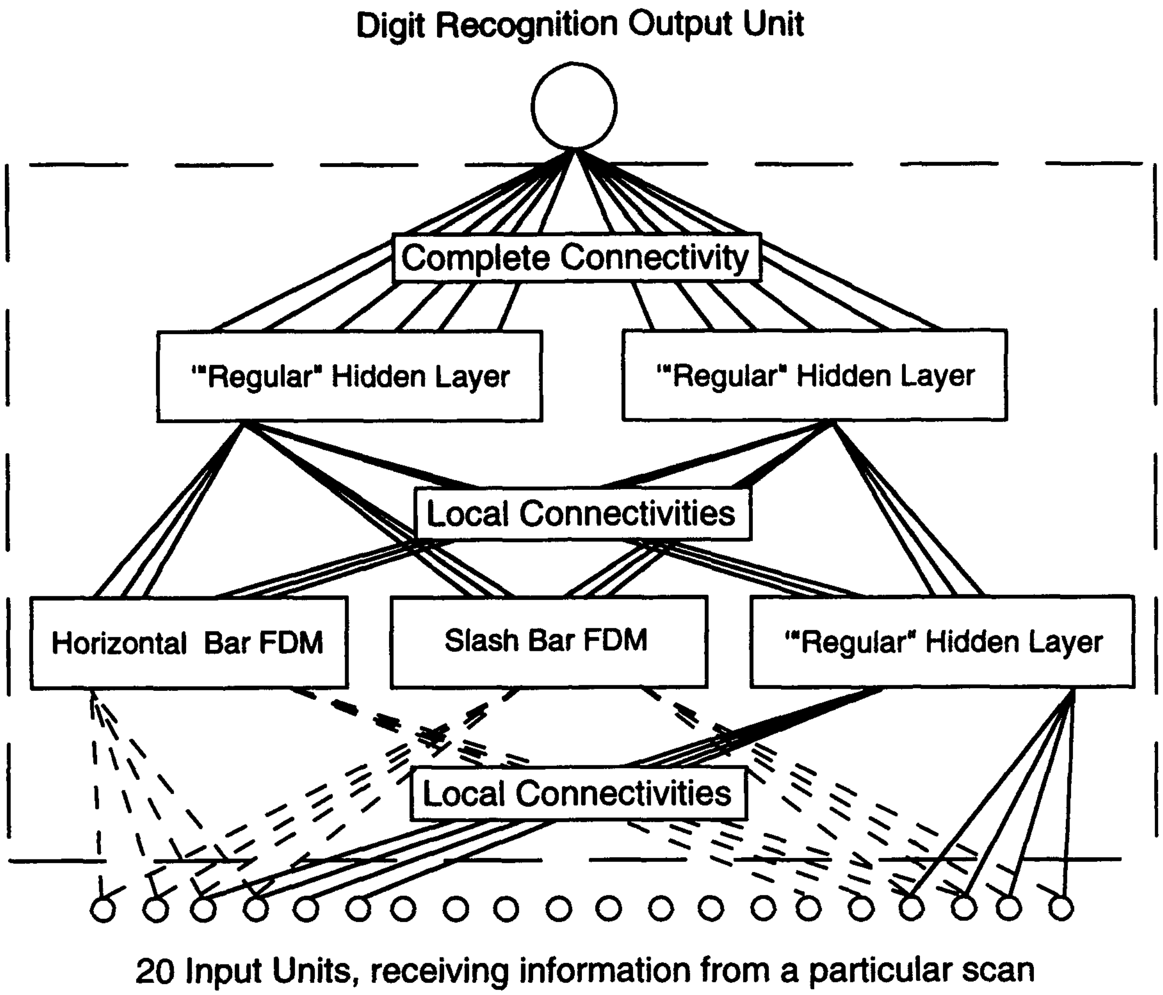


Figure 7: A Single Scan Network Module (SSN)

were connected to the output unit using links with delays of 1, 3, 5, and 7. Self-recurrent and threshold unit links were placed on all units. This SSN consisted of 161 units and 1,490 links.

Single Digit Recognition Networks

Consider scanning the image of an isolated digit using a left-to-right column-wise scan. Although useful discriminatory information may be present in the rightmost columns of the image, this information is not detected by the network until the final time steps. Consequently, it may be more effective to employ multiple scans in a variety of directions, where each scan feeds information into a separate group of input units. Use of multiple scans also adds a degree of redundancy, and hence, robustness to the recognition process.

In the multiple-scan situation, information from each scan is processed independently and concurrently by the SSNs associated with each scan and the output of each SSN is passed to a single output unit. This complete network is referred to as a *Single Digit Recognition Network* (SDRN), an example of which is shown in Figure 8. 80 input units are used in this case, aligned in 4 banks of 20, which receive information from

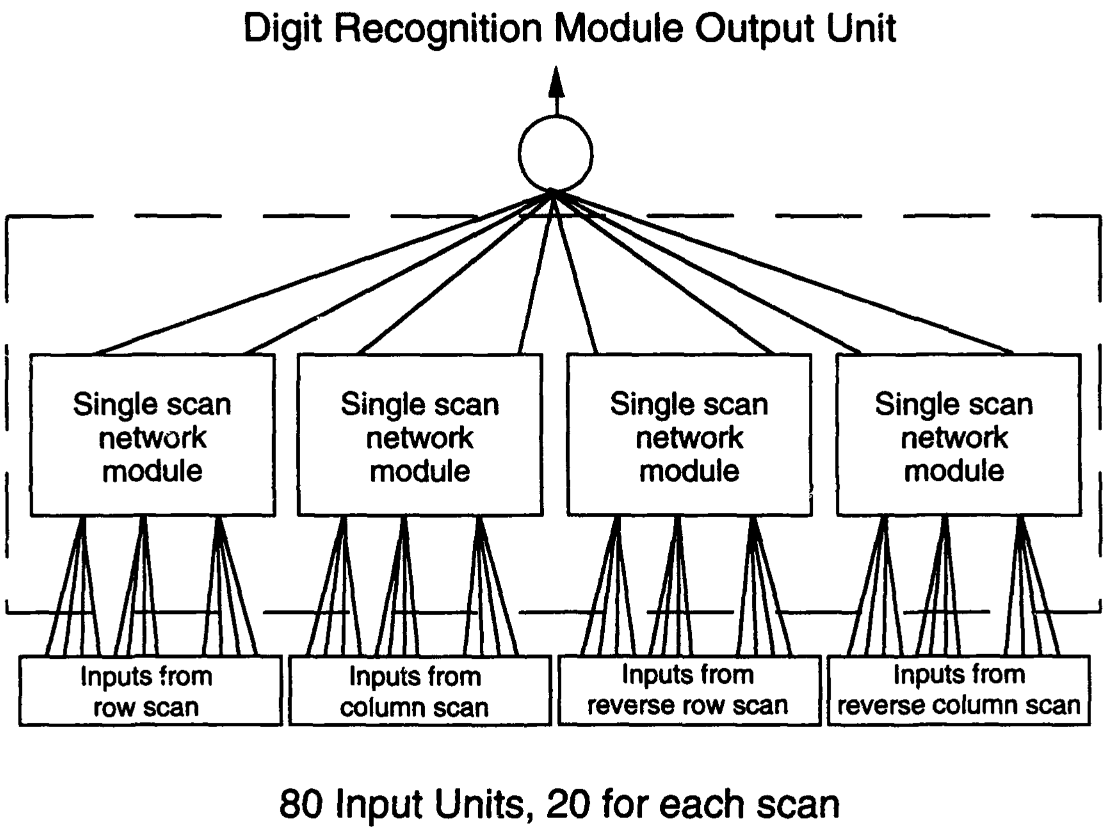

Digit Recognition Module Output Unit
Single scan network module
Single scan network module
Single scan network module
Single scan network module
Inputs from row scan
Inputs from column scan
Inputs from reverse row scan
Inputs from reverse column scan
80 Input Units, 20 for each scan

Figure 8: A Single Digit Recognition Network Module (SDRN)

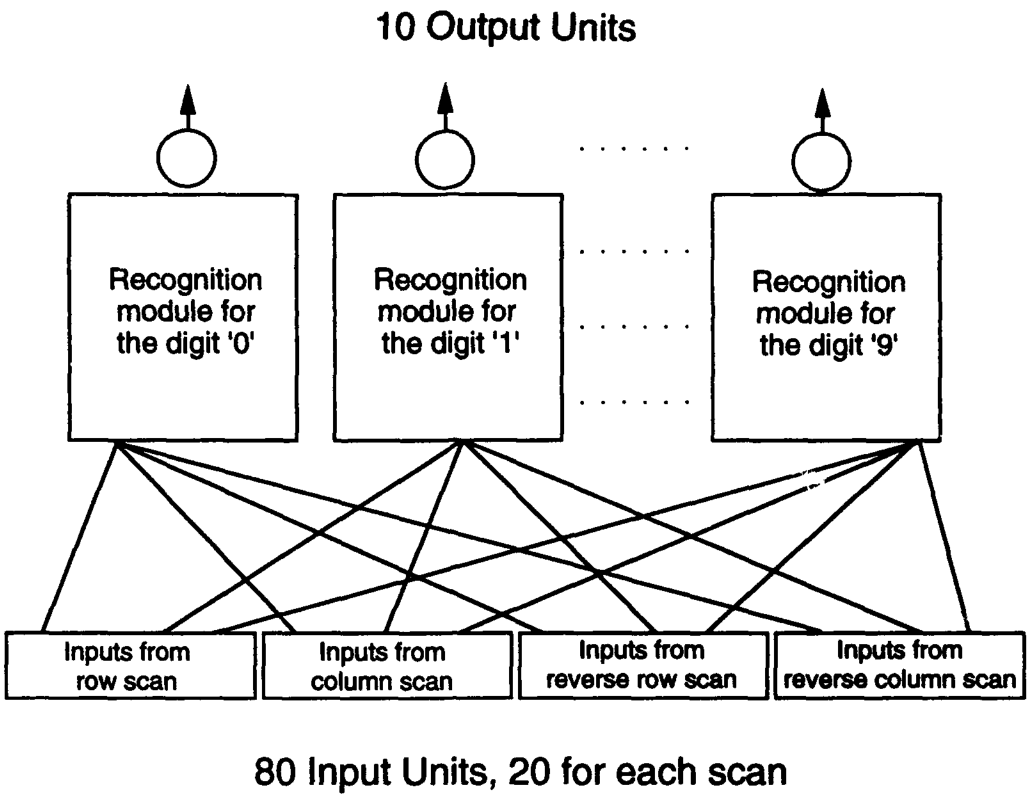

10 Output Units
Recognition module for the digit '0'
Recognition module for the digit '1'
Recognition module for the digit '9'
Inputs from row scan
Inputs from column scan
Inputs from reverse row scan
Inputs from reverse column scan
80 Input Units, 20 for each scan

Figure 9: A Refined Recognition Device (RRD)

4 scans. Information from each scan is processed independently in separate SSNs, and the information is combined at the output level. The dashed box delimits the entire Single Digit Recognition Module. Each Single Digit Recognition Network is trained to recognize a single digit class, and reject all others.

The Complete RRD

After each Single Digit Recognition Module is trained to recognize its respective digit, all networks are combined to produce the RRD, capable of recognizing all ten digits. Figure 9 depicts an RRD that uses four scans.

3.3 Coarse Recognition Device (CRD)

The Coarse Recognition Device is designed to provide coarse character recognition, in the form of hypothesis formulation, and to estimate inter-character segmentation points based on the available evidence. The CRD architecture is a special case of the RRD architecture in which only one Single Scan network is used within a Single Digit Recognition Network. This network receives information from a left-to-right scan. As scanning progresses and more of the image is viewed, confidence in digit classifications is updated. At each time step, the CRD generates a signals for all confidences exceeding a thresholds. The CRD therefore produces coarse recognition estimates. If only one character is present in the image, the CRD produces a signal after it has observed enough of the character to recognize it. The multi-character case is similar, except that CRD signals are also interpreted by the PC as hypotheses for inter-character segmentation points.

A specific CRD used in our experiments possessed the following characteristics: 6 Feature Detection Modules (FDM) in the first hidden layer contained 9 LRFs each (the LRFs had the same structure as before). The second hidden layer was arranged in 6 banks of 6 units, with each bank receiving input from a corresponding FDM. Each unit in a bank received information from 4 contiguous LRFs via unit delay links. The units in the second layer of the Single Scan Networks were connected to the output unit using links with delays of 1, 3, 5, and 7. Self-recurrent links were placed on all units. The complete CRD network consisted of 111 units and 1,118 links.

3.4 Procedural controller (PC)

A traditional component, the Procedural Controller, is used to control system flow, incorporate systematic domain knowledge, and make final classification decisions.

The PC passes each connected component in the image to the connectionist recognition modules. For each component, it monitors the output of the CRD as it assimilates the component in a left-to-right fashion and waits for the CRD to build up recognition confidences. When one or more thresholds are met, the PC sends the most recently scanned portion of the image to the RRD for verification. If the RRD accepts a singular hypothesis, a digit is recognized, the CRD is reset to a zero state, and the system continues scanning to recognize the next digit. If the RRD rejects the estimate, however, the CRD must either continue processing, or backtrack. For example, if a continued scan increases confidence in the current hypothesis, it is again sent to the RRD for verification. If a continued scan decreases confidence, then thresholds can be altered to be less pessimistic and a portion of the image rescanned.

Our current implementation of the word recognition system uses little domain-specific knowledge. This was for two reasons. First, the purpose of the implementation was primarily to develop the spatio-temporal connectionist components and benchmark their base discriminatory capabilities. Second, a substantive amount of work has been done on incorporating domain knowledge into word recognition (Doster, 1977; Riseman and Hanson, 1974; Shingal and Toussaint, 1979). The following discusses some potential uses of domain knowledge that would be easy to incorporate in our system design.

Typically, a ZIP code consists of either 5 or 9 digits. This knowledge can be used by the PC to maintain a running estimate of how many digits remain to be seen, and use this estimate to guide the segmentation and recognition process. Second, although it is common to have an overlapping pair of digits, it is rare to have overlapping triplets. Furthermore, the frequencies of two consecutive overlapping digits varies greatly depending on the class of each digit. Figure 10 (adopted from (Fujisawa et al., 1992)) depicts the frequencies of touching pair combinations. That is, given a randomly sampled set of touching pairs, the left graph

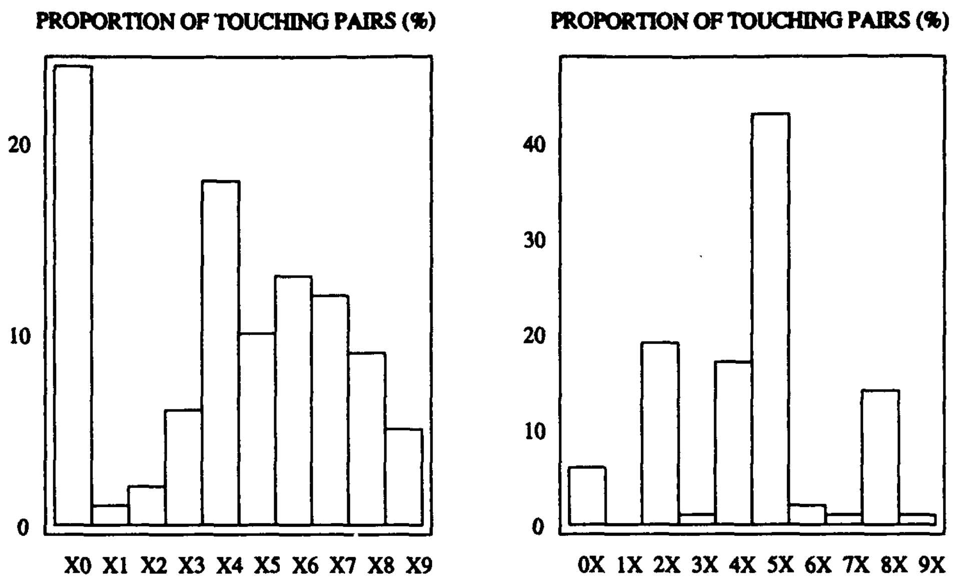


Figure 10: Frequencies of touching hand-printed digit pairs

shows how frequently each digit class can be expected to be in the trailing digit position. The right graph depicts how frequently each digit class can be expected to be in the leading digit position. The PC can utilize such knowledge when integrating the signals emanating from the RRD and CRD. In many hand-print domains, only a subset of all possible strings are legal and hence, a dictionary of legal strings can be made available. This permits the utilization of predictive dependencies between characters, derived from statistical analysis of the dictionary (eg, (Bledsoe and Browning, 1959)) and the usage of contextual word postprocessing algorithms (eg, (Doster, 1977; Shingal and Toussaint, 1979)).

The incorporation of such domain knowledge is fairly straightforward when a procedural component is used. In particular, our approach allows the PC to interact with the connectionist networks *during* recognition, making knowledge-driven recognition possible.

4 Training and testing methodology

4.1 Datasets

A good dataset for hand-printed digit and word recognition should be widely available and voluminous, with the number of authors approaching the number of images. Furthermore, the authors should be from a diverse background, and be unaware that their printing will be used to train and test a recognition device. The "United States Postal Service Office of Advanced Technology Handwritten ZIP Code Database (1987)" meets all these requirements and was made available for research by the Office of Advanced Technology, United States Postal Service. The database contains approximately 2,400 grayscale images of hand-printed five and nine digit ZIP codes, scanned from letters passing into the Buffalo, New York, Post Office. To simplify bookkeeping, only five digit ZIP code images were used. Next, the images were converted from grayscale to binary images. Finally, each ZIP code image was broken down into five individual digits by making linear slices between consecutive digits, without removing stray marks or extended strokes.

A second database containing pre-segmented hand-printed characters, the NIST Special Database 3, was made available for research by the National Institute of Standards and Technology. The database contains 313,389 images of isolated alpha-numerals, including 223,125 digits, drawn from a multi-authored set of 2,100 images of full-page hand-printed forms.

4.2 Division of Datasets

The USPS database was used for both training and testing the RRD and CRD, and testing the word recognition system. The database contains ZIP code images with serial numbers ranging from bd_0001 to bd_2636.[6] Prior to viewing the database, ZIP codes with serial numbers from bd_0001 to bd_2000 were designated as training images, while ZIP codes with numbers between bd_2001 and bd_2636 were designated as test images. The training set consisted of 1,090 five digit ZIP code images. Of the 617 ZIP code images set aside for testing, only 540 were eventually used. 59 images were excluded because they were 9 digit ZIP codes. One image contained only 4 digits and was not used, while another which was incorrectly coded was also discarded. Another 16 images contained dark lines running across them due to postal marks and scanner anomalies and were discarded. This division yielded 5,450 isolated ZIP code digit images for use in training the RRD and CRD, 2,700 isolated digit images for use in testing the RRD and CRD, and 540 complete ZIP code images for testing the word recognition system. Figure 11 illustrates the first 90 ZIP code images in the test set.

In addition to the above, a set of approximately 16,000 digit images, for use in training the RRD and CRD, was randomly sampled from the 223,125 isolated digit images in the NIST database. The remaining set of 207,000 images were reserved for testing the RRD and CRD.

Training set for RRD

The RRD is expected to reject all images which do not contain isolated digits. Since this includes non-digit blobs, it was necessary to incorporate such images into the training set as negative examples. Therefore, additional training data for disjoint strokes was synthesized. The RRD is also expected to reject components containing multiple digits. Therefore a dataset containing multiple digits was created. Finally, the CRD may signal a recognition hypothesis *before* it has completely observed the leftmost digit. Consequently, a portion of the component containing only a partial digit may be sent to the RRD for inspection. The RRD should

[6]Images from bd_1001 to bd_1500 are stored on a separate tape and designated by the USPS as test images for complete ZIP code recognition systems. These images were not used.

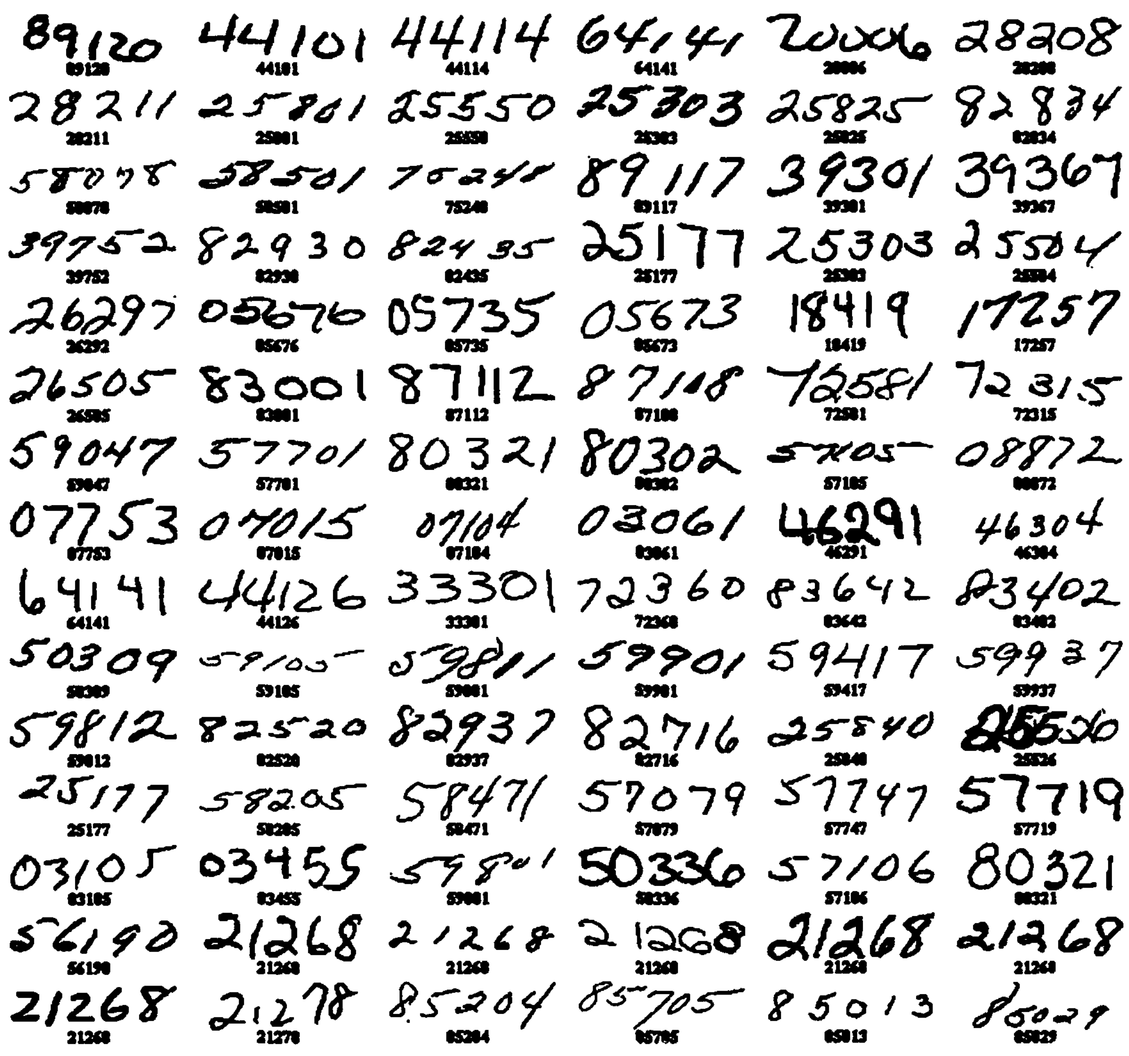


Figure 11: First 90 ZIP codes in the USPS test set

reject such incomplete digit images and accept more fully formed digits. In view of this another dataset containing partial digits was constructed..

Synthetic Data for Disjoint Strokes and Partial Images: A total of 44 images containing pieces of broken "5"s and 405 images containing partial digits were synthesized as negative training instances. To produce the 44 "5" pieces, 22 images of broken "5"s were selected from the USPS training set. The digit "5" as chosen because it seems to be the most common digit printed using multiple strokes. A visual inspection of a sample of 500 ZIP codes revealed a total of 76 instances containing a broken "5", as opposed to only 2 instances of a broken "4".

To produce the 405 partial digit images used to simulate partial data which might be provided by the CRD in the form of premature conjectures, the following steps were taken: (i) For each digit (except 1), a set of 45 images containing the digit was randomly sampled from the USPS training set. For each of the 405 images, a block of contiguous columns on the right hand side of the image was deleted. A random number of columns ranging from 33% to 50% of the total image width were removed. Finally, a completely blank image was also included to form a set of 450 negative examples of disjoint strokes and partial images.

Synthetic Data for Multiple Digits: A set of 500 images of overlapping digit pairs was synthesized. For each of the 100 possible digit pair orderings XY (eg, 01, 02, ... , 99), 5 images were generated as follows: (i) two digits, X and Y, was randomly sampled, with replacement, from the USPS training set, (ii) the digit images were separately skew-normalized and scaled to a uniform height, (iii) the X and Y images were horizontally juxtaposed to form a single image (some images were further "squashed" by a random amount between 0% and 10% of their width to simulate large overlaps), and (iv) the XY image was then scaled to fit in a 20x20 image, preserving the aspect ratio, and skeletonized. 450 of these images were retained in the negative training set.[7]

The Complete RRD Training Set: For each Single Digit Recognition Network, a total of 2025 positive and 2025 negative training examples were used. For the positive instances, 425 samples were drawn from the USPS dataset, and 1600 samples were drawn from the NIST dataset. For negative instances, 125 images of each digit class (other than the class being learned) were used in conjunction with 450 partial images and 450 multiple digit images. Of the negative examples of digits of the class not being learned, 45 were randomly sampled from the USPS set and 80 were sampled from the NIST set.

Training set for CRD

Unlike the RRD, the CRD need not be explicitly concerned with rejecting images of partial or multiple digits. Consequently, no synthesized images of partial or multiple digits were required as negative examples in the CRD training. Earlier experiment, however, had demonstrated the propensity for a single scan network

[7]In future work, real-world samples should be collected for training, instead of resorting to synthesizing data. This would not only provide a more realistic sample of overlaps, but also would be reflective of the distribution of the types of overlaps.

to focus on strokes parallel to the scanning direction. To offset this effect images containing a horizontal stroke across the entire image, were used as negative examples. In addition, the empty image and 9 images containing a sparse number of random dots were used as negative examples.

In addition to the above, the USPS and NIST data used to train the RRD was also used to train the CRD. Thus 2025 images were used as positive single digit recognition examples (425 USPS and 1600 NIST images) and 2055 images were used as negative examples. These consisted of 225 examples of each other digit (112 from USPS, 113 from NIST), the empty image, 20 images of horizontal strokes, and 9 random dot images.

4.3 Data Representation

Two methods of representing characters for hand-print recognition devices are typically employed: feature-level and pixel-level. In feature-level representation, features such as strokes or edges of various orientations are extracted from an image. The set of features is decided a priori and/or through automatic selection from a set of large pool of features pool using, for example, information-theoretic measures. In pixel-level representation, a system operates directly on the pixels of images. Images are however, typically preprocessed to remove noise and normalize certain types of variations. In general, the structural (visual) integrity of the character in an image is retained throughout the preprocessing stage. Pixel-level representation forces a learning method to acquire the features necessary for discrimination. In our work we made use of pixel-level input representations.[8]

Preprocessing

Preprocessing an image can enhance the recognition capabilities of a system by normalizing certain variations. The following pre-processing steps were methods used on isolated digit images.

Low pass filtering: High frequency noise, or "pepper" noise, commonly occurs in imaging. To reduce the adverse effects of pepper noise, a mask, shown in Figure 12, was convolved across each image. The convolution smoothed the image, and subsequent binarization eliminated stray pixels. The binarization threshold was set such that on-bits were converted to off-bits unless the weighted sum of the mask exceeded 1/2.

Skew normalization: One source of variance produced by differences in handedness and style is the *skew* of print. Skew in hand-print can be viewed as a distortion produced by an author favoring, and perhaps elongating, strokes in certain orientations which (often systematically) deviate in angle from prototypical stroke orientations. A moment-based transformation to correct individual character skew (Bakis et al., 1968) was applied to all isolated digit images. This technique is equivalent to shifting rows of bits horizontally to remove the skew.

[8]The use of pretrained feature detectors within the network can however, be thought of as a "hybrid" of pixel and feature based representations.

1/16	1/8	1/16
1/8	1/4	1/8
1/16	1/8	1/16

Figure 12: Low pass filter mask used to remove pepper noise

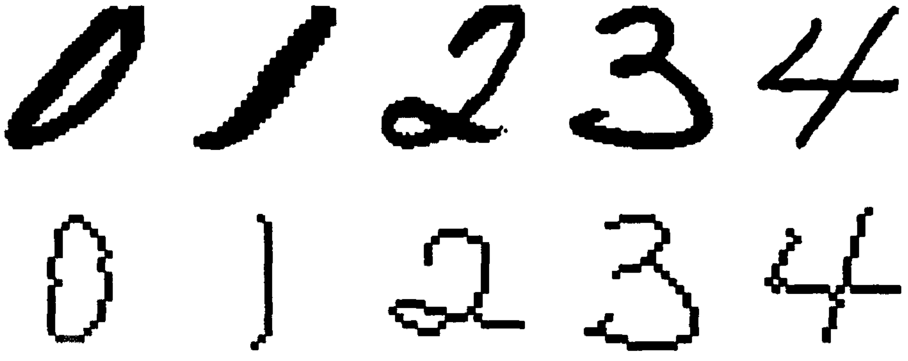


Figure 13: Examples of ZIP code digit images before and after preprocessing

Size Normalization; Isolated digit images for training the RRD were encased in a bounding box by trimming off surrounding white space and scaled down to a 20x20 image. The aspect ratio of the character was preserved by padding with white space, if necessary. A nearest neighbor method which sampled pixels in the original image at regular intervals was used to perform the scaling (Hou, 1983). In the case images used for training the CRD, the image was scaled down to fit in a rectangle containing twenty rows of pixels while preserving the aspect ratio. The number of columns therefore varied, depending on the width of the digit. After scaling, the image was skeletonized. This scaling routine is used for CRD since it processes arbitrary long inputs.

Skeletonization: Skeletonization was employed to remove variation caused by differing thicknesses of writing styli and image quantizations. Skeletonization erodes pixels from a binary image until strokes of only a single pixel width remain. The SPTA skeletonization method (Naccache and Shinghal, 1984) was used.

Examples of images before and after filtering, deskewing, scaling, and skeletonization are show in Figure 13. The digits in the top line are the original images and the digits on the bottom line are the digit after preprocessing.

4.4 Target Functions

Since spatio-temporal networks generate an output at each time over the assimilation of the input, one must specify, for each training example, the desired (target) activity of output units at each step of processing. Several classes of target functions were considered. These included linear, step, Gaussian, and sigmoid functions. Based on our experience, the following asymmetric sigmoid target was used: the target value at the first time step for both positive and negative examples was 0.05. At subsequent times t, the target value for positive examples followed a rising sigmoid curve, while the target value for negative examples stayed constant at 0.05.[9]

Intuitively, for positive examples, the confidence in a particular classification should increase slightly with each time step near the onset of the image. By the midrange of the image, or slightly thereafter, enough information should have been assimilated to classify the image with some confidence. By the end of assimilation, the network should be certain that it has seen a particular character class.

Since digits of a particular class may contain instances of widely varying widths and heights, it may be useful to tailor the target functions to individual examples. In the case of the RRD, digits were centered in a 20x20 image for recognition. A fixed sigmoid target function was found to be adequate for positive examples. Since multiple orthogonal scans were utilized, at least one single scan network was able to receive image information to satisfy a fixed target.

Since the CRD uses only one scan and needs to assimilate images of differing widths, it cannot use a homogeneous target function over all examples. Furthermore, the CRD must possess shift-invariant characteristics, allowing it to ignore arbitrary amounts of white space before reaching the onset of a digit. Therefore, the target function was chosen to be a sigmoid with its onset, inflection point, and duration customized for each example. To enforce shift invariance, the left side of each positive example was "padded" with a random amount of white space (from 1 to 30 contiguous columns). The target response during the area of white space was 0.05. The target response during assimilation of the actual digit was a sigmoid, rising from 0.05 at the onset to 0.95 at its end, with its inflection point placed 60% through the extent of the example.

4.5 Training

Training was done using the second-order quasi-Newton Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm (Luenberger, 1984) using (i) GRADSIM — a system for applying nonlinear gradient optimization techniques to train spatio-temporal connectionist networks from examples (Watrous, 1988) and (ii) GRAD-CM2 a data-parallel version of GRADSIM implemented on a Connection Machine CM-2 (Fontaine, 1992). In general, runs were terminated when (1) MSE had fallen below 0.0025, and (2) error reductions were insignificantly small over a large number of objective function and gradient evaluations.

[9]The output values lie in the interval [0,1].

4.6 Network Scoring

The following methods were used to make classification decisions, based on the output of the network over time.

Integrated Activation

Since the output unit was trained to respond with increasing activity upon presentation of an positive example, and respond with non-increasing activity to negative examples, a simple "unit score" based upon integrated activation was used. The score for a given output unit was determined by summing the individual activations at each time step over assimilation of the entire image, and then normalizing by the extent of the image.

In the case where one output unit was employed per class to be recognized (eg, the RRD), a classification decision was made using a simple winner-take-all approach, wherein the image is classified as belonging to the class corresponding to the output unit which generated the largest time-normalized integrated output.

The integrated outputs of the units can be interpreted as probability estimations by normalizing the values to obey the laws of probability (Bridle, 1990). Although this transformation does not affect single object classification (in a winner-take-all sense), it is useful in the integration of various components of a recognition system (eg, the RRD, CRD, and PC). The statistical properties of the underlying written language can more readily be integrated, and communication between the CRD and RRD can be viewed as joint probabilities, as opposed to suggestive signals. Normalization to estimate probabilities was used in the word recognition system.

Rejection Criterion

It is often of practical importance to assess the performance of a recognition system by deriving the percentage of test images that must be rejected as unclassifiable in order to achieve a lower error rate on the remaining images. Consequently, a rejection criterion was defined. Considering time-normalized integrated activation, let A_h be the highest activation of the N output units, and let A_s be the second highest activation. A measure of classification confidence, C, was defined to be $C = (1 - A_s)/(1 - A_h)$.

Since $A_h, A_s \in (0,1)$, and $A_s \leq A_h$, we have $C \geq 1$. Larger values of C indicate more confident classifications. The rejection criterion was then defined such that for some $\epsilon > 0$, if $C < (1 + \epsilon)$, then the image was rejected as being unclassifiable.

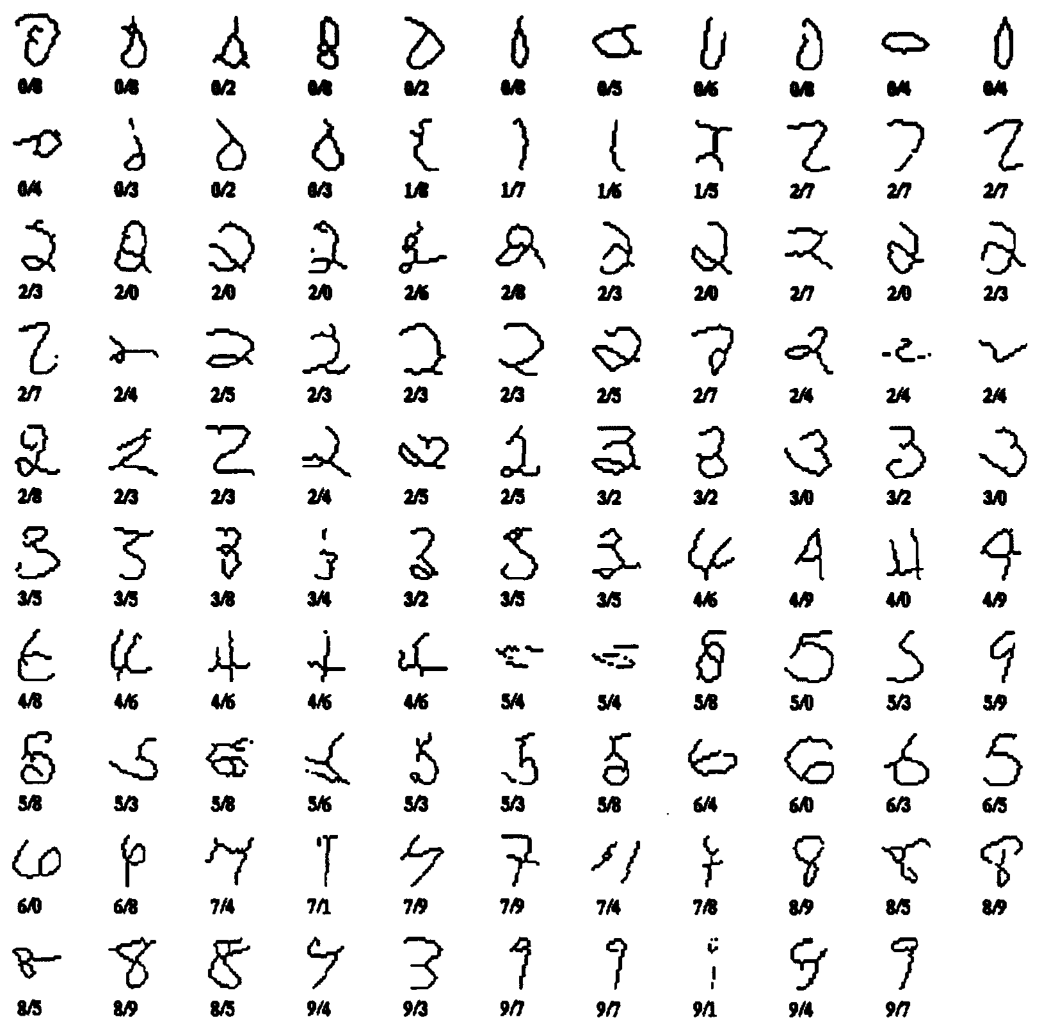


Figure 14: USPS test images misclassified by the RRD

5 Results

5.1 Refined recognition device (RRD)

On the NIST test set of 207,000 isolated digit images, an accuracy of 96.5% was achieved at a 0% rejection rate. On the USPS test set of 2,700 images, an accuracy of 96.0% was obtained with no rejections. All USPS test set misclassifications are shown (in preprocessed form) in Figure 14. The number before the slash below each image is the true classification and the number after the slash is the (incorrect) RRD classification. The USPS accuracy is comparable to other reported results using test samples drawn from the USPS database (eg, (Denker et al., 1989; Knerr et al., 1992; Le Cun et al., 1990)). Although caution must be taken in making comparisons since true performance measures cannot be obtained without using identical test databases, visited only once by each recognition system.

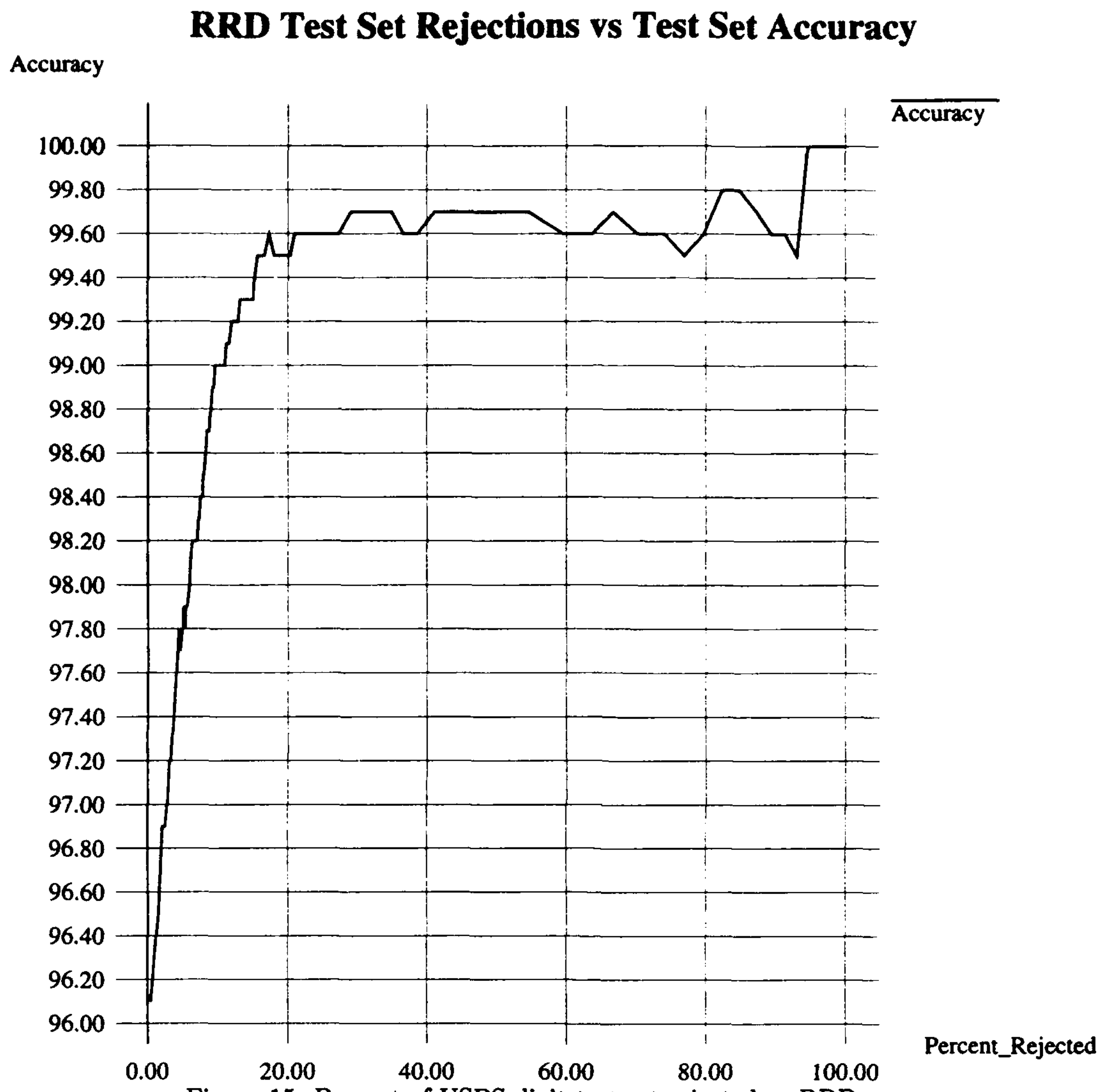


Figure 15: Percent of USPS digit test set rejected vs RRD accuracy

In real-world implementation, a user is often willing to allow a system to reject a portion of samples as being unclassifiable in exchange for improved accuracy. The performance of the RRD, as an increasingly larger percentage of the USPS test images are rejected, is shown in Figure 15. The rejection criterion used was based on the ratios between the highest and second highest time-integrated unit activations. Figure 15 was derived by incrementing a rejection threshold, ϵ (cf. Section 4.6), until all images were rejected. The steep rise in accuracy with a small number of rejections is highly desirable, and a 99% accuracy was obtained upon rejecting 9.5% of the images. A detailed analysis of the RRD performance may be found in (Fontaine, 1993).

Threshold	Case 1	Case 2		Case 3		Accuracy
Value	Incorrect	Correct	Incorrect	Correct	Incorrect	
0.4	48	720	48	1746	138	91.3%
0.5	62	335	13	2149	141	92.0%
0.6	68	136	10	2354	132	92.2%

Table 1: CRD hypothesis results on the USPS digit test set

5.2 Coarse recognition device (CRD)

The CRD was trained on single digits in order to enable it to make a good hypothesis concerning the first digit it encounters as it scans an image in a left-to-right fashion; the focus was not on constructing a CRD capable of stand-alone recognition of digits. Consequently, the CRD was evaluated on images containing single digits (the USPS set of 2,700 digits) using two modes of operation. The first mode measured its base recognition capabilities. The second mode was geared towards inspecting the ability of the CRD to formulate hypotheses using a simple threshold method.

In the first mode, classification was performed by choosing the output unit which yielded the highest time-normalized integrated activation. This allowed for variations in the width of each digit without explicitly changing the target function for each test image. The aim was to evaluate the base discriminatory capability of the network for isolated digit recognition. On the USPS set of 2,700 digits, the CRD achieved an accuracy of 94.4% with no rejections. Samples in error are depicted in Figure 16 (for ease of viewing, the samples are shown scaled to fit in a bounding box).

The second mode of operation was geared towards evaluating the capability of the CRD to produce hypotheses. Note that the formulation of classification hypotheses is different from that of segmentation point hypotheses (which is detailed in Section 5.3). The same dataset was used, but instead of making a classification based on integrated activation (after an entire image is assimilated), a classification was made when any output unit activation exceeded a predetermined threshold. After a classification was made, the CRD was reset and scanning continued. Although the first classification produced is of primary interest, resetting the network and continuing the scan helped gauge the robustness of the CRD to ignore partial images. The decision process in the second mode of operation resulted in three possibilities for classification: (1) no output ever exceeded threshold, and hence no classification was made, (2) a classification was made, the network was reset, and one or more other classifications were subsequently made, and (3) exactly one classification was made. Case (1) is undesirable in the context of the overall system. Case (2) is acceptable if the first digit recognized is the actual digit being scanned. Likewise, case (3) is acceptable, if the classification is correct. Table 1 shows the CRD results using various output unit threshold values. The accuracy was computed by summing the Case 2 and Case 3 Corrects and dividing by the total number of images (2700).

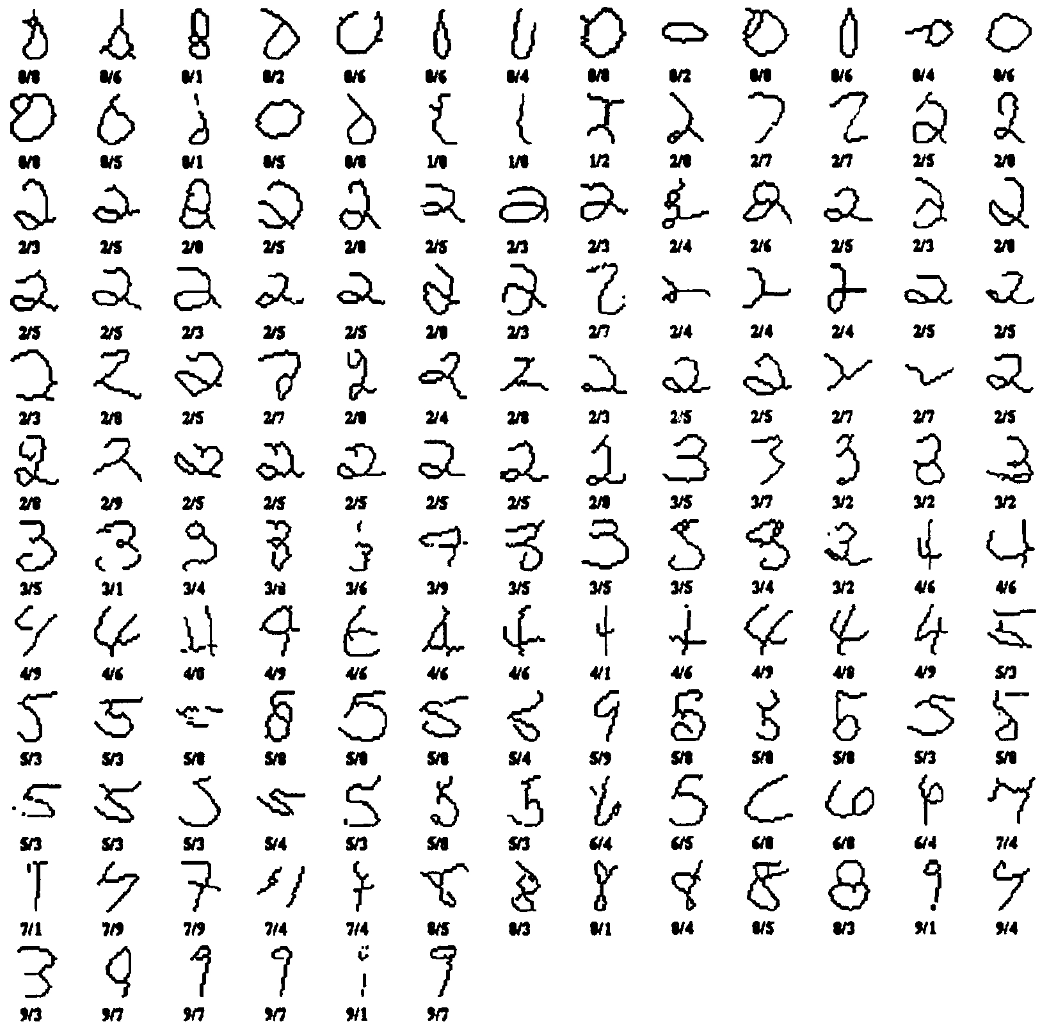


Figure 16: USPS digit test images misclassified by the CRD

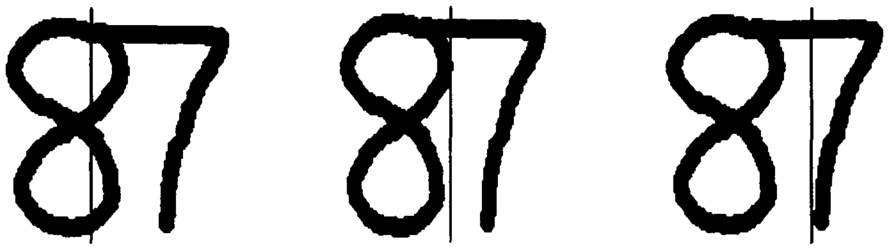


Figure 17: Examples of premature, good, and late segmentation points

The results were as expected, with more rejections occurring with higher thresholds, accompanied by fewer multi-digit classifications. Also, first digit classification accuracy seems acceptable. A detailed analysis of the RRD performance may be found in (Fontaine, 1993).

5.3 Word recognition system

Before presenting the performance results we outline the functioning of the word recognition system as currently implemented and discuss the generation of segmentation points based on the CRD response.

Threshold Derivation

The following questions need to be addressed in the spatio-temporal framework: At what point during the CRD inspection of an image is there enough evidence to hypothesize a classification? How much of the image should be sent to the RRD, i.e., where should a segmentation be made? Ideally, a hypothesis should be made as soon as possible during the inspection of an image in order to expedite recognition and the segmentation point should correspond to the end of the digit being assimilated.

Figure 17 depicts three examples of hypothesized segmentation points. The leftmost example shows a premature segmentation point, the center example shows a good segmentation point, and the rightmost example shows a late segmentation point. Hypothesizing an early segmentation point has two disadvantages. First, system throughput is decreased since extra interaction must occur between the CRD and RRD to refine the segmentation point. Second, if the RRD accepts a premature segmentation point, the CRD is forced to examine the remaining portion of the digit and may posit "ghost" hypotheses. For example, suppose a segmentation point is hypothesized by the CRD roughly 75% through assimilation of the digit "8", as in the leftmost example in Figure 17. If the RRD accepts the hypothesis the CRD is forced to examine the remaining 25% of the image and may hypothesize the presence of a "3" (the RRD was explicitly trained to reject partial images for exactly this reason). If a segmentation point is positioned too late, such that it overruns the next digit, both the RRD verification of the current digit and the CRD assimilation of the next digit can be affected. Fortunately, there exists a simple method for producing fairly good segmentation points.

Using the target function for positive instances during CRD SDRN training as a model for future response, the width of a test digit being assimilated may be estimated at any point during recognition. The target function used for positive examples during SDRN training was a sigmoid:

$$d(x) = \frac{1}{1 + e^{-(M \times (x - C))}} \tag{1}$$

where $d(x)$ is the desired output of the SDRN when a given fraction, x, of the image had been assimilated ($x \in [0,1]$). M is a (positive real) value controlling the shape of the sigmoid ($M = 10.0$ was used for the RRD and CRD), and C is a fixed fraction of the length of the target specifying the location of the inflection point. If C is 0.5, the output unit response will exceed 0.5 after half of a positive example is assimilated. If it is assumed that positive test examples will respond according to this target, then half of a digit's width will have been witnessed by the network when its output unit activation exceeds 0.5. This width may be doubled to serve as an estimate of the digit's actual width and, consequently, a segmentation point. Although this scheme does not produce exact segmentation points for all cases, it was found to approximate the point reasonably well.

In addition to projecting the segmentation point, one also needs to determine the level of output activation at which the CRD should hypothesize the presence of a digit. The current implementation uses a simple threshold method and makes a hypothesis when a CRD output unit exceeds a predefined threshold, θ. The choice of θ should be large enough to reduce false positives, yet small enough to allow the CRD to make enough conjectures. In all experiments, θ was set to be 0.5 (the target value at the inflection point). In future work, we plan to use a dynamic value of θ which is derived automatically based on performance results on an appropriate set of training data.

Processing by the word recognition system

Given a binary image containing one or more digits, recognition progresses in three stages: (i) a component recognition stage in which the RRD tries to identify whether any connected components are well-formed digits, (ii) a rejected component analysis stage in which the CRD and RRD interact to classify the remaining components of the image, and (iii) a decision making stage to assign a classification and confidence to the image as a whole.

Stage 1: Connected Component Recognition

Connected components are found and each connected component in the image is passed to the RRD. The RRD acceptance criterion is set pessimistically, since it is desired to recognize only those components which can confidently be recognized as digits. The threshold was set such that the RRD recognized isolated digits with a 99.5% accuracy, rejecting 16.8% of the images. At the end of Stage 1, the RRD acceptance threshold was altered to be more accepting in Stage 2. The threshold was set to obtain 99.0% accuracy at a 9.5%

rejection rate.

After the components are sent to the RRD, the skew of each recognized component is weighted by its mass, and an average measure of skew, μ, is produced. The remaining components are deskewed by a factor of μ. In cases where no component was recognized (which rarely occurred), the image is not deskewed.

Stage 2: Rejected Component Recognition

The components not accepted in Stage 1 by the RRD are inspected by the CRD in Stage 2. One or more components are joined if there is not significant columnar white space between them (where "significant" is taken to be a fraction of the height of the image—15% in the current implementation).

The CRD then processes each image component separately. During left-to-right assimilation, if any CRD output unit exceeds the threshold value of $\theta = 0.5$, a hypothesis is made and a projected segmentation point is computed. The image area between the last accepted segmentation point (or the image onset) and the hypothesized segmentation point is sent to the RRD for verification. If the RRD rejects the hypothesis, the CRD continues. If it accepts, the segmentation point is moved forward until RRD confidence decreases. The classification and confidence are recorded, and both the CRD and RRD networks are reset. If the CRD produces no hypothesis during assimilation of a component, it is forced to provide its most confident single digit classification, regardless of the confidence level.

At this point, a classification for the entire image can be reported. Stage 3, however, combines the evidence from the individual classifications to produce an overall confidence level.

Stage 3: Decision Making

In the current implementation, recognition of each digit in the image is taken to be independent of the other digits. Since each classification produces a confidence level expressible as a probability, a classification probability is assigned to the entire image by multiplying the probabilities associated with each digit classification. Thus, the only action taken in Stage 3 is a simple multiplication of probabilities. It is considered a separate stage, however, since algorithms taking into account the underlying distributions can easily be employed not only to produce more confident classifications based on available domain knowledge, but also to produce ranked hypotheses concerning missing or extra digits (Doster, 1977; Riseman and Hanson, 1974; Shingal and Toussaint, 1979).[10]

[10]In Stage 2, the CRD is operating in "forced" mode. In "unforced" mode, if the CRD either cannot produce a hypothesis while assimilating a component, or if the CRD and RRD cannot agree, a "don't know" value is produced. The cited algorithms can be used to instantiate the "don't know" values, based on information produced during recognition and the class conditional probabilities.

Set	Drawn From	% Overlap	% Touching	% First Correct	% Pair Accuracy
1	Train	0	9.6	95.6	87.8
2	Train	5	46.6	92.6	77.0
3	Train	10	63.8	92.8	66.2
4	Test	0	10.2	95.0	87.6
5	Test	5	45.6	92.6	74.0
6	Test	10	59.8	92.0	65.6

Table 2: Recognition results on synthesized pairs of USPS digits

5.4 System Results

Results on the problems of overlapping digit pair recognition and USPS ZIP code recognition are now presented. The PC utilized no domain knowledge regarding individual character form, frequency, or contextual dependencies. In addition, no restrictions were assumed on the number of digits which could appear in an image, the amount of white space (or lack of white space) between consecutive digits, or author-specific style. The goal was to evaluate the base discriminatory capabilities of the system without relying on domain knowledge and heuristics. One underlying assumption, however, was that adjacent characters showed, to some extent, uniformity in their baselines, skew of print, and size. This is not an overly restrictive assumption for most hand-print recognition tasks, since authors tend to print uniformly within a word.

Digit Pair Recognition

The capability of the system to segment and recognize overlapping and touching digit pairs was tested. Since pairs of digits which touch, or whose fields overlap, are not readily available, test data was synthesized from the isolated USPS digit images. Images of digit pairs were generated as explained in Section 4.2. The system was tested on 6 separate data sets. Each data set was comprised of 500 images, with 5 images of each possible XY combination of the 10 digits. The 6 sets differed depending on whether the digits were drawn from the training or testing set, and how much they overlapped. Table 2 shows recognition results with the CRD in forced mode, rejecting no images. The columns of the table represent, respectively, the test set number, the USPS set from which the digits were drawn (train or test), the overlap percentage used during pair synthesis, the percentage of the set containing touching pairs as a result of the overlap, the percentage of the set in which the first digit of the pair was correctly classified, and the percentage of the set in which the pair was correctly classified. A pair classification was deemed correct if and only if both digits were correctly classified. Figure 18 depicts the images in Set 6, the most difficult test set, which were correctly identified.

The percentage of each set containing digit pairs which touch is significant. It is common for traditional

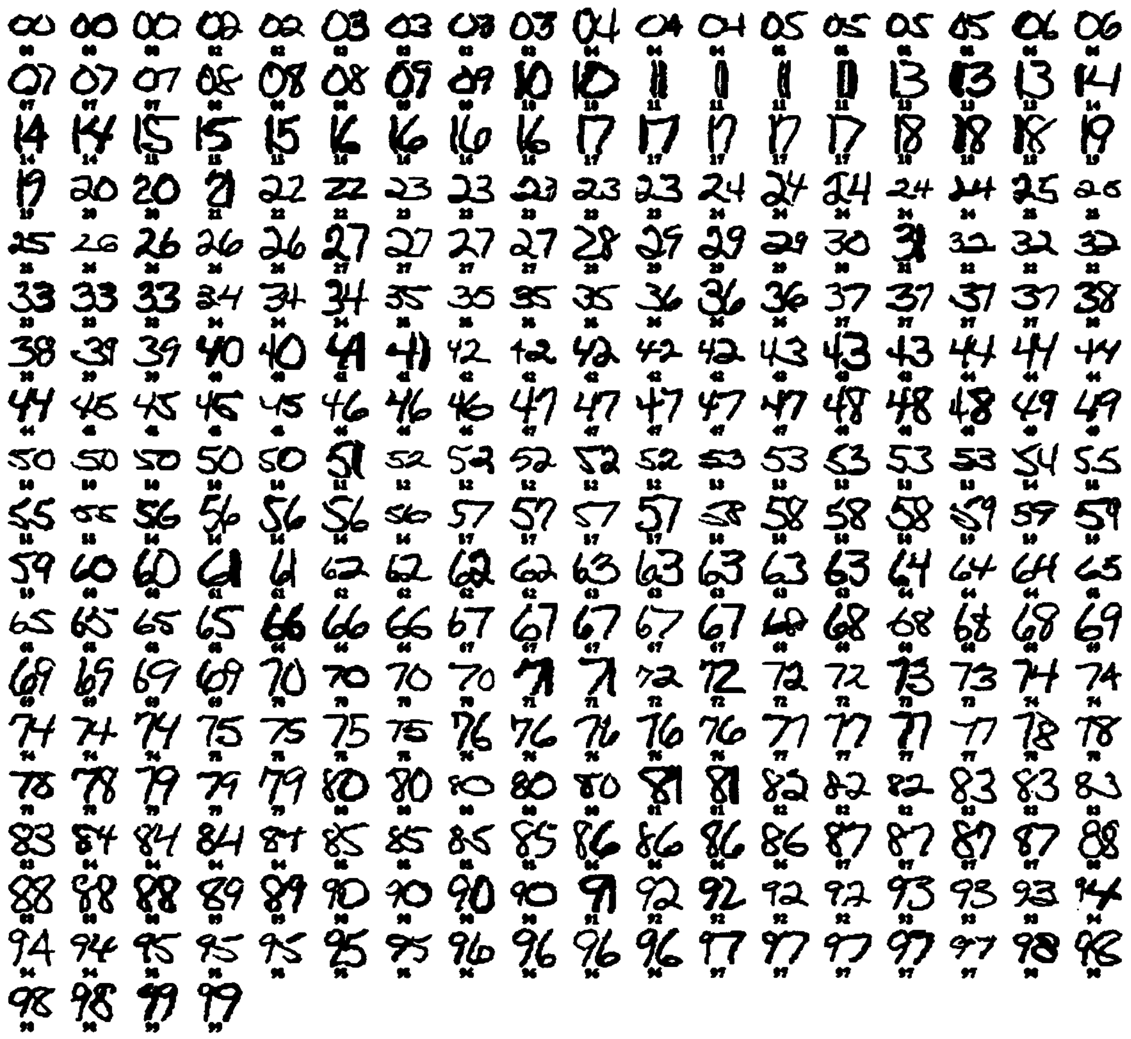


Figure 18: Correctly identified USPS digit pair images from Set 6

segmenters to utilize columnar whitespace to hypothesize segmentation points. Yet, since this test data (by construction) contains no inter-character white space, many segmenters would experience difficulty in dealing with such samples.

In addition, the percentage of the set in which the first digit of the pair was correctly classified is also important. If the first digit can be classified with good confidence, which the results suggest, the classification could be used to help disambiguate subsequent digits, particularly if the class conditional distributions are known. One could also imagine dual CRDs operating conjointly, one assimilating data from a left-to-right scan and the other from a right-to-left scan. Their classifications could be compared, with more weight placed on the first classification of each and less weight on subsequent classifications.

It is difficult to draw performance comparisons to other approaches due to a lack of a standardized test sets and a dearth of reported results on digit pair (and string) classification. Moreover, many results which are reported are alphabet-dependent, relying on topological features of the digits to provide hints for segmentation points. These results, however, are comparable to another reported result using synthesized sets of digit pairs without alphabet-specific knowledge (Keeler et al., 1991). In addition, Table 2 shows little variation between images created from the training or test sets, suggesting good generalization.

The ability of the system to recognize digit pairs was further tested by assuming it was known that exactly two digits were present in each image. If such knowledge were available at the onset of recognition, a system could be tailored to perform more effectively. Here, the assumption was made to facilitate error analysis. After the system classified an image, if the classification was not exactly two digits long, the image was considered to be rejected.

Table 3 summarizes recognition results, allowing the system to reject classifications not of length 2. It depicts the distribution of images rejected due to their length (1 or 3 digits long), the percentage of test set classification rejected due to length, and the system accuracy on the remaining images. Figure 19 illustrates all rejected and incorrect classifications. Rejected classifications are prefixed with "R:" in their label. The true classification appears before the slash in the label of an image, and the system classification appears after the slash.

As expected, a significant increase in accuracy is achieved. More importantly, however, the results indicate that the system is being too pessimistic. This is evidenced by the high ratio of the number of rejects of length 1 to the number of rejects of length 3 and suggests a future area of work.

ZIP Code Recognition

The same system which was applied to digit pair images was also applied to the real-world ZIP codes provided by the United States Postal Service. The system was able to correctly classify 66.0% of the 540 test images. A classification was deemed correct if and only if it matched the true ZIP code exactly. Note that the 66%

Set	Rejected Image Length		% Total Rejects	% Accuracy
	1	3		
1	39	8	9.4	96.9
2	98	5	20.6	97.0
3	149	9	31.6	96.8
4	28	9	7.4	94.6
5	95	12	21.4	94.1
6	138	9	29.4	92.9

Table 3: Recognition results on synthesized pairs of USPS digits with length rejection

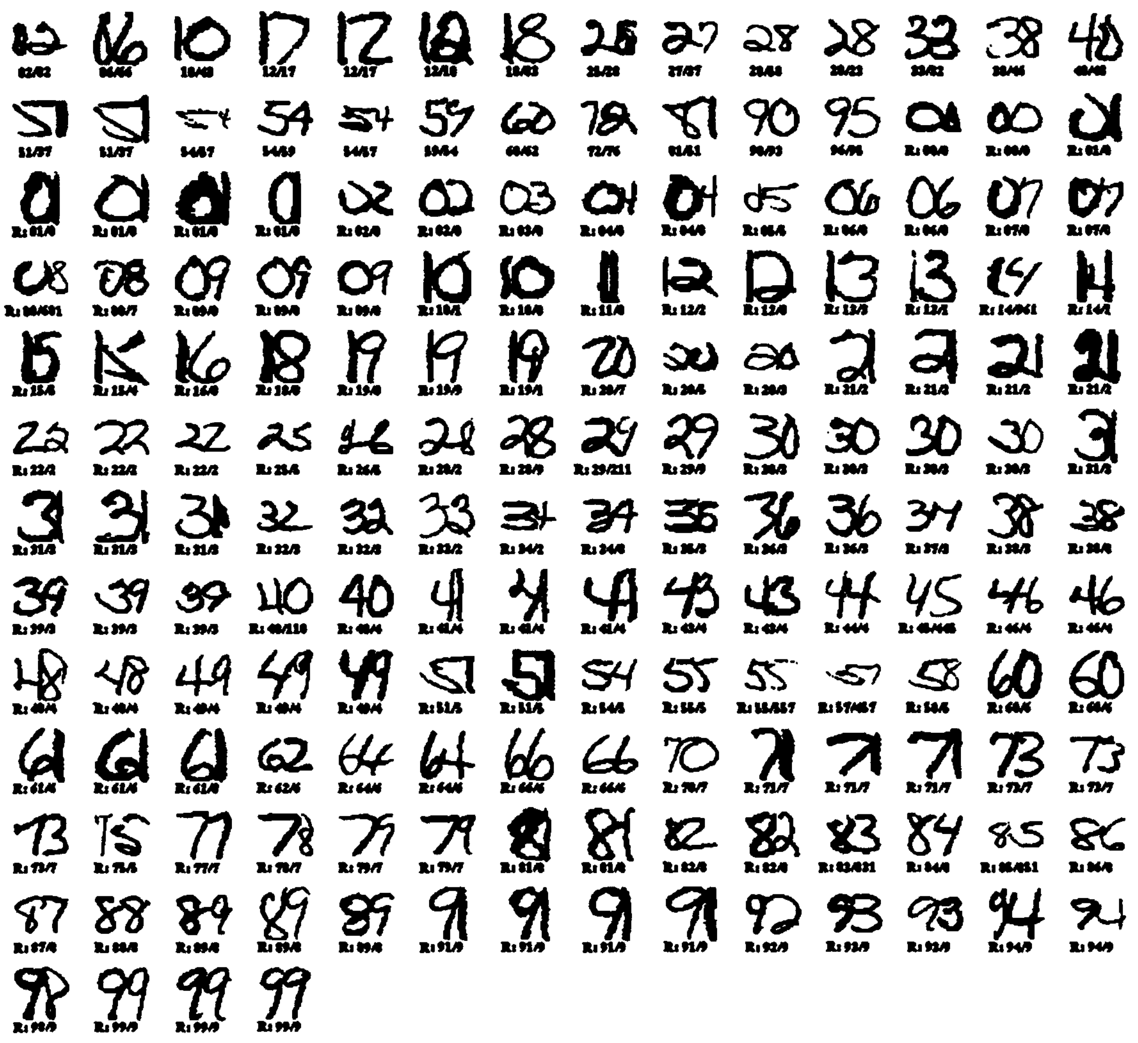


Figure 19: Rejected or incorrectly identified USPS digit pair images from test Set 6

Length of Classification	1	2	3	4	6	7
Number of Occurrences	1	2	10	57	22	4

Table 4: Frequencies of ZIP code rejections due to classifications not of length 5

rate is therefore a "worst-case" measurement—it considers a classification of an entire ZIP code incorrect in the event that any constituent digit is incorrect.

For the same reasons as digit pair recognition, it is difficult to make performance comparisons for ZIP code recognition. One benchmark for comparison, however, is the accuracy of the RRD if it were able to inspect each digit in a ZIP code as if it were an isolated digit. Since the RRD achieved a 96.0% accuracy on the USPS test set of isolated digits, it can be expected to correctly classify approximately $100 \times 0.96^5 = 81.5\%$ of the ZIP codes, assuming the digits were correctly isolated. Of course, this is an upper bound (given the described RRD), and the system cannot be expected to achieve such accuracy for several reasons. A significant number of the ZIP codes contained touching sequences of digits, disjoint digits, stray blotches, and ascenders/descenders from other lines on the envelope. More exactly, the set of 540 images contained 97 overlapping digit pairs, more than 80 disjoint digits, several stray blotches, and 17 ascenders/descenders. The system, as implemented, can hardly hope to classify an image containing a stray blotch or an ascender (descender) correctly, since it is forced to generate an extra digit classification.

Performance was also measured by rejecting classifications which did not contain exactly five digits. Accuracy increased to 80.4% at a 17.8% rejection rate. Figure 20 shows the ZIP codes which were still classified incorrectly. The label below each image denotes the actual ZIP code (before the slash) and the system's classification (after the slash).

Table 4 reports the frequencies of the length rejections. Although 70 classification were rejected due to omission of one or more digits, only 26 were rejected on the basis of extra digits. This suggests that the either the RRD should be more accepting, the CRD should be less pessimistic, or both.

Using the confidences based on multiplying individual digit classification probabilities, Figure 21 depicts rejection rate versus accuracy on USPS ZIP codes, assuming 17.8% have already been length-rejected.

6 Discussion

In this section we discuss several issues which arose during the conception, formulation, and implementation of the spatio-temporal approach to visual pattern recognition, and outline promising avenues for future work. The problem of recognizing hand-print by machine is neither new nor solved. This investigation can be placed in proper context if one considers the research and development effort that has been spent on this problem over the past forty years. The intent of our effort was to investigate an alternate framework for

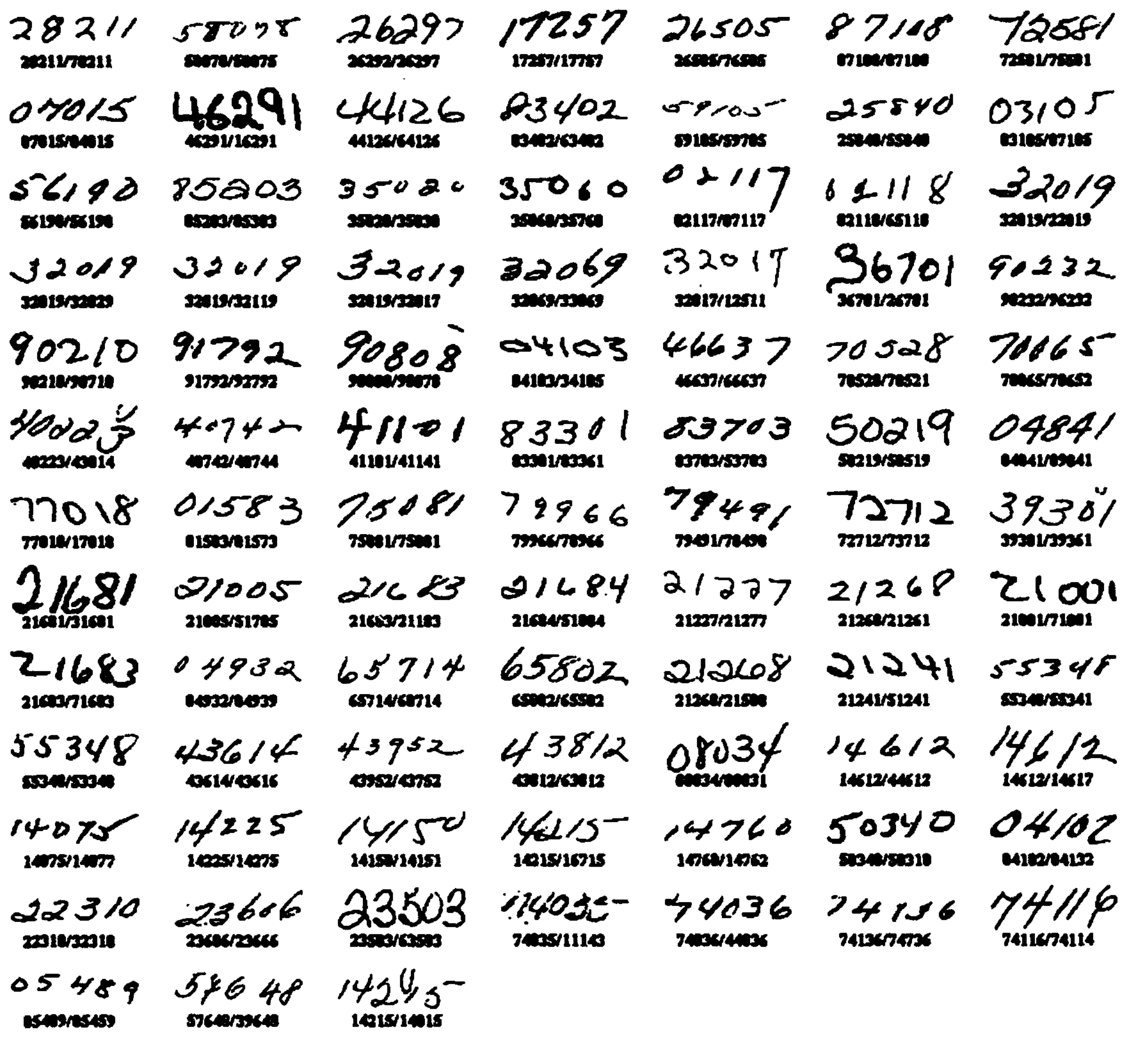


Figure 20: Incorrectly identified USPS ZIP codes

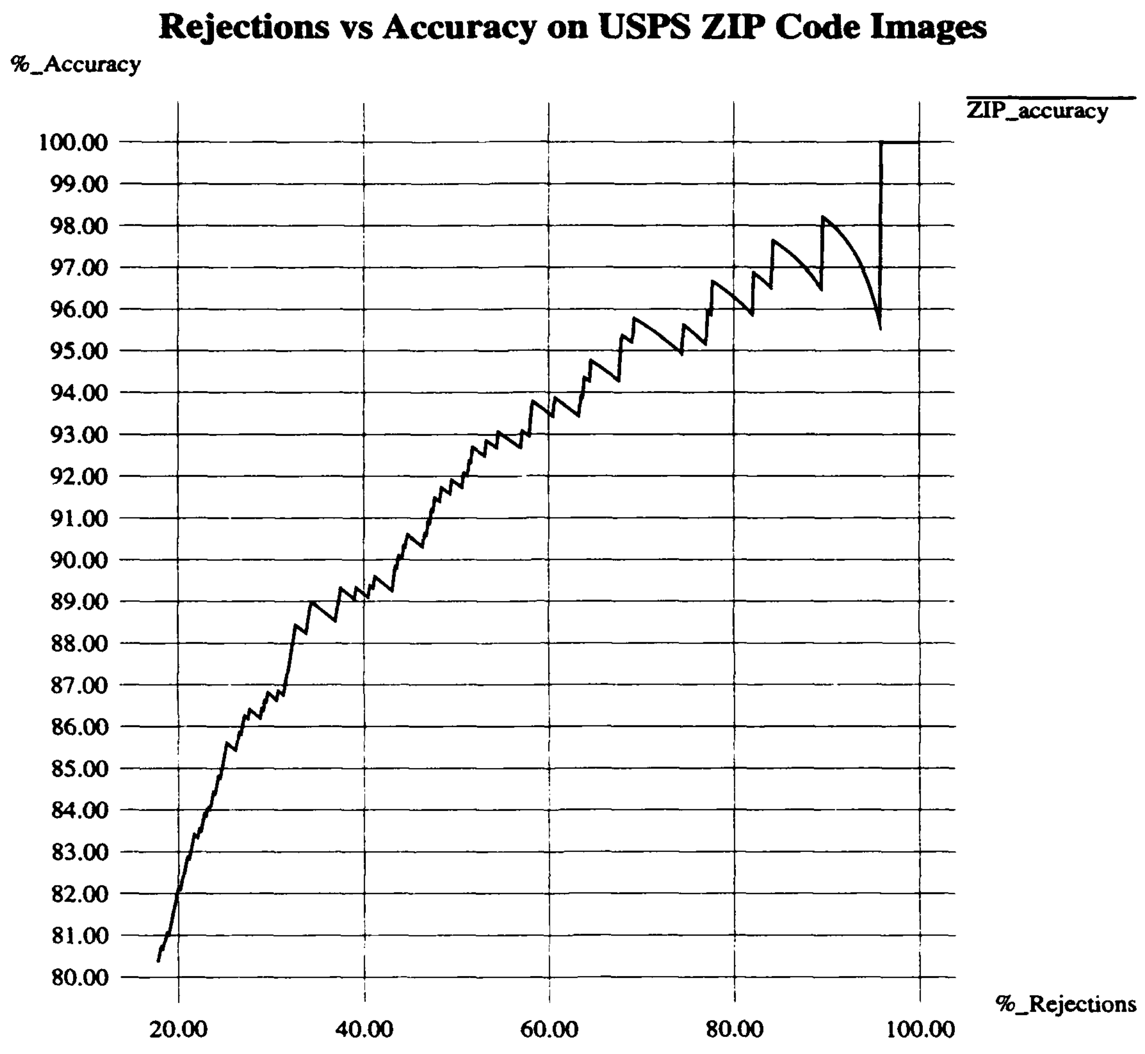


Figure 21: Accuracy vs rejections on 5 digit ZIP code classifications

coping with some of the limitations inherent in many conventional approaches.

Traditional approaches have proceeded by dividing the word recognition problem into two phases: segmentation into component characters followed by recognition of each component. The vast majority of research effort has been invested in developing devices capable of carrying out the second phase of recognizing isolated characters, with relatively modest progress made in the segmentation area. To some extent, it is not surprising that an adequate solution has remained elusive using this methodology. Given an arbitrary alphabet, and a pair of overlapping characters from that alphabet, it is simply not possible to segment the pair without using a mechanism to (partially or completely) recognize the component characters. This dilemma strongly suggests the need for the development of alternate models of recognition to cope with such cases. This investigation has taken a first step towards addressing this fundamental problem.

6.1 Validation of the Spatiotemporal Approach

Do spatiotemporal models offer advantages over traditional feedforward networks on visual recognition problems such as character recognition? In a theoretical sense, there is a type of equivalence, since a spatiotemporal network can be "unfolded" in time and viewed as a spatial network. Structurally, however, emulating a spatiotemporal network in a feedforward sense involves multiple replication and concatenation of network structures, depending on the extent of the examples to be assimilated. Thus, from an optimization and implementation viewpoint, the equivalence is quite tenuous.

Our experience suggests that the spatiotemporal approach has several advantages over feedforward networks: shift-invariance and explication of local image geometry along the temporalized axis, a reduction in the number of free parameters occurs, and the ability to process arbitrarily long inputs. The latter is particularly relevant in the context of hand-print recognition, since it provides a natural mechanism within the connectionist framework to cope with the segmentation/recognition dilemma.

Validation through empirical investigation, however, ultimately relies on the produced results. On just the problem of isolated digit recognition, the utility of the approach was verified, evidenced by recognition results which are comparable to the current state of the art on a real-world set of difficult digit images. Further, it was seen that spatiotemporal networks are capable of recognizing images of multiple and overlapping digits. Good recognition accuracy was achieved on difficult images which many traditional segmenters could not possibly segment and recognize. Although several areas for improvement were discovered through analysis, and extensions are possible, the presented results substantiate the approach and warrant further investigation.

6.2 System Formulation

Refined Recognition Device

Ideally, the RRD should recognize characters in the target alphabet and reject all non-character blobs. Of course, the dimensionality of the input space makes the derivation of such a device very difficult. From a system development point of view, however, it suffices to construct an RRD capable of recognizing characters and rejecting non-character blobs *which it is likely to encounter.* This point of view was taken into consideration during RRD development for digit recognition. Within the context of the envisaged system, it was decided that the RRD should be capable of rejecting multiple and partial digits, since it was likely to encounter these cases.

By setting CRD hypothesis parameters conservatively, it was hoped that it would be better to venture premature segmentation point hypotheses rather than late hypotheses, allowing the CRD/RRD interaction to derive a suitable segmentation point. Thus, negative training instances of "partial multiples" were not used. However, two problems cropped up as a result of not employing such training examples. First, although CRD parameters were set to produce conservative (premature) segmentation point estimates, late segmentation point estimates were occasionally made. Second, in the implemented system, if a hypothesis was accepted by the RRD, the hypothesis boundary was moved forward, and the process repeated until RRD confidence was reduced. This sometimes resulted, somewhat surprisingly, in an accurate segmentation point being expanded to a late segmentation point, as RRD confidence (incorrectly) did not recede, despite the fact that it was receiving a partial multiple. To a large extent, the CRD/RRD interaction relies on the ability of the RRD to make appropriate rejections. Therefore, in future work, the RRD should be trained on a sizable body of negative partial, multiple, and partial multiple images.

Coarse Recognition Device

The CRD's recognition performance fell short of the RRD's. The shortcoming was due to the combination of the increase in learning task complexity and the decrease in network mechanism. It appears that the overall performance can be improved if the target function is aligned later in the assimilation process so that the the CRD can witness a larger portion of the image before making hypotheses.

Although results on recognition of difficult digit pair images suggest that vertical segmentation produces good results on the set of digits, we need to look at other sorts of segmentation boundaries. One should not expect that a vertical slice will always segment a digit pair into relatively clean images of two (fully-formed) digits. It should also be possible to improve vertical segmentation by augmenting the RRD training set with images derived by vertically slicing multi-digit images. That is, starting with a set of overlapping digit pairs, each pair could be vertically sliced at a "good" location, and the component images used as positive training examples for the RRD.

The methodology used to train the CRD, in which isolated digits were used as training examples and positive instances were trained to respond in accordance to rising sigmoid targets, is only one of several possible approaches. An interesting alternative is to train the CRD on multiple digit images in order to explicitly enforce segmentation point hypotheses. Using target functions comprised of multiple Gaussian peaks, centered at the midrange of each digit, provides a method for the CRD to recognize pairs independently of the RRD.

Another approach is to perform a local search in the area of a hypothesized segmentation point, sending each portion to the RRD. This could correct the problem of late segmentation points. A method which seems likely to produce very good results (albeit somewhat brute-force) is to employ two CRDs, one of which assimilates images in a left-to-right fashion and the other in a right-to-left fashion. Since it was seen that the CRD was capable of quite accurately recognizing the first digit of a pair, the classifications made by each of the two CRDs could be weighted accordingly. In the single digit case, more confident classification could be made. And, problems involving inter-class similarity of localized regions could be reduced.

Finally, it should be noted that the CRD may be quite useful in combination with other segmentation approaches, since it is capable of producing a good estimate of the region of overlap between digits.

Procedural Controller

In the implemented system, the PC was deemphasized and charged primarily with simple monitoring and decision making tasks. The incorporation of procedural algorithms utilizing the statistical properties of the written language can greatly augment performance, both during and after the connectionist networks' inspection of the image. Although postprocessing algorithms have been studied and found to be effective in augmenting recognition performance (eg, (Doster, 1977)), they are of less interest here, since they may easily be added to any recognition system.

The usage of character distributions and other domain knowledge *during* processing is an advantage offered as a consequence of the spatiotemporal approach. As the CRD is assimilating an image, for example, more of the spatial structure of the image is revealed. As classification confidences build, one could imagine interpreting the confidences with respect to the statistical distributions of the language, thereby affecting CRD hypotheses. In addition, procedural algorithms could be interjected before assimilation is complete, in order to verify hypotheses or refine segmentation points.

6.3 Extensions

The extension of the approach to other character sets is of great practical interest. Consider an N class recognition problem, in which the goal is to classify an object as belonging to one of N classes. Assuming a forced classification, a total of $\frac{N(N-1)}{2}$ inter-class boundaries exist. The extension from the set of 10 digits

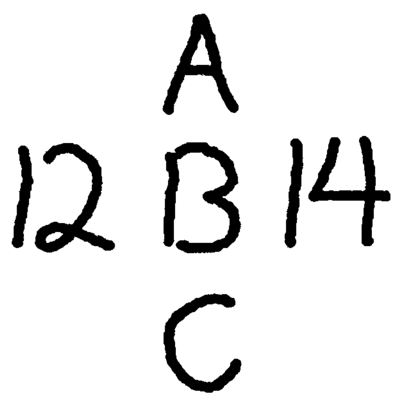


Figure 22: An example of the role of context

to the set of 36 digits and lower case letters, for example, results in an increase from 45 to 630 inter-class boundaries.[11] Hence, any recognition system is faced with a more difficult recognition task as the number of classes increases.

Both the number of character classes and the geometric forms of the character classes contribute to the difficulty of recognizing overlapping characters. As illustrated in Figure 22, recognizing certain characters, or sequences of characters, is not always possible. Due to the high-level nature of context affects involved, such cases are of limited interest. However, similar examples requiring inspection of localized areas are necessary for recognition and are of interest. Due to the nature of the basic CRD scheme, in which an image is assimilated in a left-to-right fashion over time, certain combinations of adjacent character can produce "ghost characters", requiring assimilation beyond the true segmentation point in order to disambiguate the pair. For example, consider overlapping a lower case "c" (on the left) with a lower case "r" (on the right). The "cr" sequence contains a "ghost image" of an "a" (or possibly an "o") and as the CRD scans in a left-to-right fashion, it is necessary to progress beyond the true segmentation point in order to witness enough of the "r" in order to disambiguate the sequence. Further, what if the language in question contains characters whose overlaps cannot be adequately segmented via vertical strokes?

These concerns are justified, insofar as they question the ability of the implemented system to operate on other alphabets. The hand-printed digit recognition system described in detail in this paper, however, is only one possible instantiation of the general framework. Alternate mechanisms, some of which were described earlier in this section (eg, dual using CRDs or training the CRD to respond in Gaussian peaks, etc), can be used to augment the shortcomings of the system, as implemented. In the simplest case, the CRD provides valuable information (eg, good first character recognition rates) and can be integrated with other methods.

[11]This number is slightly less, of course, if similar character classes (eg, the letter "o" and the number "0"), are considered as one. And, it is possible (albeit unlikely) that additional classes induce only trivial (easily derived) inter-class boundaries.

Conclusion

Given the success of spatiotemporal connectionist networks in hand-print recognition, it is likely that they can be applied successfully to other visual recognition problems. Initial investigation into using spatiotemporal networks to recognize simple objects has demonstrated a stronger recognition dependence on object vertices than on edge mid-sections (Farid et al., 1993), as expected according to a theory of human perception described in (Biederman, 1985). Another difficult recognition problem, discriminating between cancerous and non-cancerous cells, is being inspected, and initial results are encouraging.

In summary, the opportunities for future research in the application of spatiotemporal connectionist networks to hand-print recognition, as well as to other visual recognition domains, are plentiful. The described advantages of the approach, combined with its demonstrated success to hand-print recognition, warrant further investigation.

Acknowledgments

This work was funded by NSF grant MCS-83-05211, ARO grants DAA29-84-9-0027 and DAAL03-89-C-0031 and ONR grant N00014-93-1-1149. The authors would like to thank Ray Watrous whose work on speech recognition influenced this work and whose software tools were indispensable; Gary Herring and John Hull for the USPS database and John Geist for the NIST database; and the Penn CIS Computer Staff for providing excellent support.

References

Bakis, R., Herbst, N., and Nagy, G. (1968). An experimental study of machine recognition of hand printed numerals. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):119–132.

Biederman, I. (1985). Human image understanding: Recent research and a theory. *Computer Vision, Graphics, and Image Processing*, 32:29–73.

Blackwell, K., Vogl, T., Hyman, D., Barbour, S., and Alkon, D. (1992). A new approach to hand-written character recognition. *Pattern Recognition*, 25:655–666.

Bledsoe, W. and Browning, I. (1959). Pattern recognition and reading by machine. In *Proceedings of the East Joint Computer Conference*, pages 225–232.

Boulard, H. and Morgan, N. (1994). *Connectionist Speech Recognition: A Hybrid Approach.* Kluwer Press.

Bridle, J. (1990). Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. In Touretzky, D., editors, *Advances in Neural Information Processing Systems*, volume 2, pages 211–217. Morgan Kaufmann.

Burr, D. (1988). Experiments on neural network recognition of spoken and written text. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36:1162–1168.

Carr, C. and Konishi, M. (1990). A circuit for detection of interaural time differences in the brain stem of the barn owl. *Journal of Neuroscience*, 10(10):3227–32.

Caskey, D. and Jr, C. C. (1973). Machine recognition of handprinted characters. In *Proceedings of the International Joint Conference on Pattern Recognition*, pages 41–49.

Chow, C. (1962). A recognition method using neighbor dependence. *IRE Transactions on Electronic Computers*, 11:683–690.

Cottrell, G. and Metcalfe, J. (1991). EMPATH: Face, gender, and emotion recognition using holons. In Lippman, R., Moody, J., and Touretzky, D., editor, *Advances in neural information processing systems 3*, pages 564–571. San Mateo: Morgan Kaufmann.

Denker, J., Gardner, W., Graf, H., Henderson, D., Howard, R., Hubbard, W., Jackel, L., Baird, H., and Guyon, I. (1989). Neural network recognizer for hand-written ZIP code digits. In Touretzky, D., editors, *Advances in Neural Information Processing Systems*, volume 1, pages 323–331. Morgan Kaufmann.

Doster, W. (1977). Contextual postprocessing system for cooperation with a multiple-choice character-recognition system. *IEEE Transactions on Computers*, pages 1090–1101.

Duda, R. and Fossum, H. (1966). Pattern classification by iteratively determined linear and piecewise linear discriminant functions. *IEEE Transactions on Electronic Computers*, 15:220–232.

Elman, J. (1990). Finding structure in time. *Cognitive Science*, 14:179–212.

Farid, H., Provan, G., and Fontaine, T. (1993). A neural net investigation of vertices as image primitives. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, pages 417–421.

Fenrich, R. and Krishnamoorthy, S. (1990). Segmenting diverse quality handwritten digit strings in near real-time. In *Proceedings of the USPS Advanced Technology Conference*.

Fontaine, T. (1992). GRAD-CM2: a data-parallel connectionist network simulator. Technical Report MS-CIS-92-55, University of Pennsylvania.

Fontaine, T. (1993). *Handprinted Word Recognition Using a Hybrid of Connectionist and Procedural Methods*. PhD thesis, University of Pennsylvania.

Fujisawa, H., Nakano, Y., and Kurino, K. (1992). Segmentation methods for character recognition: From segmentation to document structure analysis. In *Proceedings of the IEEE*, volume 80, pages 1079–1092.

Fukushima, K., Miyake, S., and Ito, T. (1983). Neocognitron: a neural network model for a mechanism of visual pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 13:826–834.

Gader, P., Forester, B., Ganzberger, M., Gillies, A., Mitchell, B., Whalen, M., and Yocum, T. (1991). Recognition of handwritten digits using template and model matching. *Pattern Recognition*, 24:421–431.

Garris, M., Wilson, C., Blue, J., Candela, T., Grother, P., Janet, S., and Wilkinson, R. (1992). Massively parallel implementation of character recognition systems. In *SPIE Conference on Machine Vision Applications in Character Recognition and Industrial Inspection.*

Guyon, I., Albrecht, P., Le Cun, Y., Denker, J., and Hubbard, W. (1991). Design of a neural network character recognizer for a touch terminal. *Pattern Recognition*, 24(2):105–119.

Highleyman, W. (1961). An analog method for character recognition. *IRE Transactions on Electronic Computers*, 10:502–512.

Hou, H. (1983). *Digital Document Processing.* John Wiley and Sons.

Jordon, M. (1987). Attractor dynamics and parallelism in a connectionist sequential machine. In *Proceedings of the Eight Annual Conference of the Cognitive Science Society, Seattle, WA.*

Keeler, J., Rumelhart, D., and Leow, W. (1991). Integrated segmentation and recognition of hand-printed numerals. In Lippman, Moody, and Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 3, pages 557–563. Morgan Kaufmann.

Knerr, S., Personnaz, L., and Dreyfus, G. (1992). Handwritten digit recognition by neural networks with single-layer training. *IEEE Transactions on Neural Networks*, 3(6):962–968.

Lam, L. and Suen, C. (1988). Structural classification and relaxation matching of totally unconstrained handwritten ZIP-code numbers. *Pattern Recognition*, 21:19–31.

Lapedes, A. and Farber, R. (1987). Nonlinear signal processing using neural networks. Technical Report LA-UR-87-2662, Los Alamos National Laboratory.

Le Cun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., and Jackel, L. (1990). Handwritten digit recognition with a back-propagation network. In Touretzky, D., editors, *Advances in Neural Information Processing Systems*, volume 2, pages 396–404. Morgan Kaufmann.

Luenberger, D. (1984). *Linear and Nonlinear Programming.* Reading, MA: Addison-Wesley, second edition.

Martin, G. and Pittman, J. (1990). Recognizing hand-printed letters and digits. In Touretzky, D., editors, *Advances in Neural Information Processing Systems*, volume 2, pages 405–414. Morgan Kaufmann.

Mozer, M. (1989). A focused backpropagation algorithm for temporal pattern recognition. *Complex Systems*, 3:349–381.

Munson, J. (1968). Experiments in the recognition of hand-printed text: Part I—character recognition. In *Proceedings of the Fall Joint Computer Conference*, pages 1125–1138.

Naccache, N. and Shinghal, R. (1984). SPTA: a proposed algorithm for thinning binary patterns. In *IEEE Transactions on Systems, Man, and Cybernetics*, volume SMC-14, pages 409–418.

Pavlidis, T. and Ali, F. (1975). Computer recognition of handwritten numerals by polygonal approximations. *IEEE Transactions on Systems, Man, and Cybernetics*, pages 610–614.

Riseman, E. and Hanson, A. (1974). A contextual postprocessing system for error correction using binary n-grams. *IEEE Transactions on Computers*, 23:480–493.

Schenkel, M., Weissman, H., Guyon, I., Nohl, C., and Henderson, D. (1993). Recognition-based segmentation of on-line hand-printed words. In Hanson, S., Cowan, J., and Giles, C., editor, *Advances in Neural Information Processing Systems*, volume 5, pages 723–730. Morgan Kaufmann.

Schürmann, J. (1982). Reading machines. In *Proceedings of the International Conference on Pattern Recognition*, pages 1031–1044.

Shingal, R. and Toussaint, G. T. (1979). Experiments in text recognition with the modified Viterbi algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1:184–193.

Shridhar, M. and Badlerin, A. (1987). Context-directed segmentation algorithm for handwritten numeral strings. *Image and Vision Computing*, 5(1):3–9.

Suen, C., Nadal, C., Legault, R., Mai, T., and Lam, L. (1992). Computer recognition of unconstrained handwritten numerals. In *Proceedings of the IEEE*, volume 80, pages 1162–1180.

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K. (1989). Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pages 328–339.

Watrous, R. (1988). GRADSIM: a connectionist network simulator using gradient optimization techniques. Technical Report MS-CIS-88-16, University of Pennsylvania.

Watrous, R. (1990). Phoneme discrimination using connectionist networks. *J. Accoust. Soc. Am.*, 87(4):1753–1722.

Watrous, R. (1991). Context-modulated vowel discrimination using connectionist networks. *Computer Speech and Language*, 5:341–362.

Watrous, R. and Shastri, L. (1986). Learning phonetic features using connectionist networks: An experiment in speech recognition. Technical Report MS-CIS-86-78, University of Pennsylvania.

Yamamoto, K. and Mori, S. (1979). Recognition of handprinted characters by outermost point method. In *Proceedings of the International Joint Conference on Pattern Recognition*, pages 794–796.

TECHNICAL REPORT TR-94-031
INTERNATIONAL COMPUTER SCIENCE INSTITUTE
BERKELEY, CA 94704

Massively Parallel Real-Time Reasoning with Very Large Knowledge Bases: An Interim Report

D. R. Mani and **Lokendra Shastri**

International Computer Science Institute
1947 Center Street Berkeley, CA 94704
and
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104

mani@linc.cis.upenn.edu
shastri@icsi.berkeley.edu

Abstract

We map structured connectionist models of knowledge representation and reasoning onto existing general purpose massively parallel architectures with the objective of developing and implementing practical, rapid or real-time reasoning systems. SHRUTI, a connectionist knowledge representation and reasoning system which attempts to model reflexive reasoning, serves as our representative connectionist model. Realizations of SHRUTI are developed on the Connection Machine CM-2—an SIMD architecture—and on the Connection Machine CM-5—an MIMD architecture.

Though SIMD implementations on the CM-2 are reasonably fast—requiring a few seconds to tens of seconds for answering queries—experiments indicate that SPMD message passing systems are vastly superior to SIMD systems and offer hundred-fold speedups. The CM-5 implementation can encode large knowledge bases with several hundred thousand (randomly generated) rules and facts, and respond in under 500 milliseconds to a range of queries requiring inference depths of up to eight.

This work provides some new insights into the simulation of structured connectionist networks on massively parallel machines and is a step toward developing large yet efficient knowledge representation and reasoning systems.

1 Introduction

Connectionist models are fast developing into widely explored architectures for cognition and intelligence. These models use a large number of simple nodes which are profusely interconnected by direct hard wired links, carrying simple, scalar messages. Massive parallelism is an important feature of any connectionist model. Since any system that purports to model human cognition must use some form of massive parallelism if it has to react in real-time (Feldman and Ballard, 1982; Shastri, 1991; Newell, 1992), *structured* connectionist models—with their inherent parallelism and their ability to represent structured knowledge—seem to be promising architectures for high-level—or symbolic—processing. Several structured connectionist models have been proposed for rule-based reasoning, language processing, planning and other high-level cognitive processes (Barnden and Pollack, 1991). From a practical standpoint, if such systems have to be fast, efficient and usable, we will need to be able to simulate or emulate them on massively parallel platforms. From a cognitive standpoint, where our concern is to design, test and prototype connectionist models of cognition, we would require suitable platforms for implementing and experimenting with these highly parallel models. Hence mapping connectionist systems onto currently existing massively parallel architectures appears to be an avenue worth exploring.

In this report we investigate the mapping of structured connectionist models of knowledge representation and reasoning onto existing general purpose massively parallel architectures with the objective of developing and implementing practical, real-time reasoning systems. We define *rapid* or *real-time reasoning* to be reasoning that is fast enough to support real-time language understanding. We can understand written language at the rate of about 150–400 words per minute—i.e., we can understand a typical sentence in a second or two (Shastri and Ajjanagadde, 1993).

SHRUTI, a connectionist knowledge representation and reasoning system which attempts to model reflexive reasoning (Shastri and Ajjanagadde, 1993), will serve as our representative connectionist model. Efficient realizations of SHRUTI are developed on the Connection Machine CM-2—an SIMD architecture—and on the Connection Machine CM-5—an MIMD architecture.[1] We shall use the term *parallel rapid reasoning system* to designate these SHRUTI-based, massively parallel, systems that can handle very large knowledge bases and perform a large yet limited class of reasoning in real-time.

Though SIMD implementations on the CM-2 are reasonably fast—requiring a few seconds to tens of seconds for answering simple queries—experiments indicate that SPMD message passing systems are vastly superior to SIMD systems and offer hundred-fold speedups. The CM-5 implementation can encode large knowledge bases with several hundred thousand (randomly generated) rules and facts, and respond in under 500 milliseconds to a range of queries requiring inference depths of up to eight.

In addition to developing viable technology for supporting large-scale knowledge base systems, this work provides some new insights into the simulation of structured connectionist networks on massively parallel machines and is a step toward developing large yet efficient knowledge representation and reasoning systems.

Section 2 is an overview of the system described in the rest of this report. Section 3 provides a brief description of SHRUTI, our representative structured connectionist knowledge representation and reasoning system. Section 4 is a general discussion of the issues involved in mapping SHRUTI onto massively parallel machines. Section 6 deals with the design, implementation and characteristics of the SPMD parallel rapid reasoning system on the CM-5. Similar issues for the SIMD CM-2 architecture are considered in Appendix A.

2 Overview of the System

The parallel rapid reasoning system supports the encoding of very large knowledge bases and their use for real-time inference and retrieval. Toward this end, the system includes the following suite of programs and

[1]Though the CM-5 is an MIMD architecture, it can only be used in SPMD (Single Program Multiple Data) mode with current software. See Section 6.

tools:

- A parser for accepting knowledge-base items expressed in a human readable input language. The language's syntax is similar to that of first-order logic (see Appendix D).
- A preprocessor for mapping a knowledge base onto the underlying parallel machine. This involves mapping the knowledge base to an inferential dependency network whose structure is analogous to that of SHRUTI, and partitioning this network among the processors of the parallel machine.
- A reasoning algorithm for answering queries. This runs on the parallel machine and efficiently mimics the reasoning process of our connectionist models.
- Procedures for collecting a number of statistics about the knowledge base and the reasoning process. These include the distribution of knowledge base items among processors, the processor load and message traffic during query answering, and a count of knowledge base items of each type (rules, facts, concepts, etc.) activated during processing.
- A utility for generating large psuedo-random knowledge bases given a specification of broad structural constraints. Examples of such constraints are: the number of knowledge base items of each type, any subdivision of the knowledge base into domains, the ratio of inter- and intra-domain rules, and the depth of the type hierarchy.
- Several tools for analyzing and visualizing the knowledge base and the statistics gathered during query answering.

This collection of programs and tools facilitates automatic loading of large knowledge bases, incremental addition of items to an existing knowledge base, posing of queries and recording of answers, and off-line visualization and analysis of system behavior. It also allows a user to construct large artificial knowledge bases for experimentation.

The system is interactive and allows the user to load and browse knowledge bases, and process queries by issuing commands at a prompt. At the same time it is also possible to process command files and use the system in an unattended batch processing mode.

3 SHRUTI—A Connectionist Reasoning System

SHRUTI, a connectionist reasoning system that can represent systematic knowledge involving *n*-ary predicates and variables, has been proposed by Shastri and Ajjanagadde (Shastri and Ajjanagadde, 1993; Ajjanagadde and Shastri, 1991). SHRUTI can perform a broad class of reasoning with extreme efficiency. The time taken by the reasoning system to draw an inference is only proportional to the *length* of the chain of inference and is independent of the number of rules and facts encoded by the system. The reasoning system maintains and propagates variable bindings using temporally synchronous—i.e., in-phase—firing of appropriate nodes. This allows the system to maintain and propagate a large number of variable bindings *simultaneously* as long as the number of *distinct* entities participating in the bindings during any given episode of reasoning remains bounded. Reasoning in the proposed system is the transient but systematic flow of *rhythmic* patterns of activation, where each *phase* in the rhythmic pattern corresponds to a distinct *entity* involved in the reasoning process and where variable bindings are represented as the synchronous firing of appropriate role and filler nodes. A fact behaves as a temporal pattern matcher that becomes 'active' when it detects that the bindings corresponding to the fact are present in the system's pattern of activity. Finally, rules are interconnection patterns that propagate and transform rhythmic patterns of activity.

SHRUTI attempts to model reflexive reasoning over a large body of knowledge. SHRUTI has been extended in (Mani and Shastri, 1993) to effectively reason with a less restricted set of rules and facts and enhance the system's ability to model common-sense reflexive reasoning.

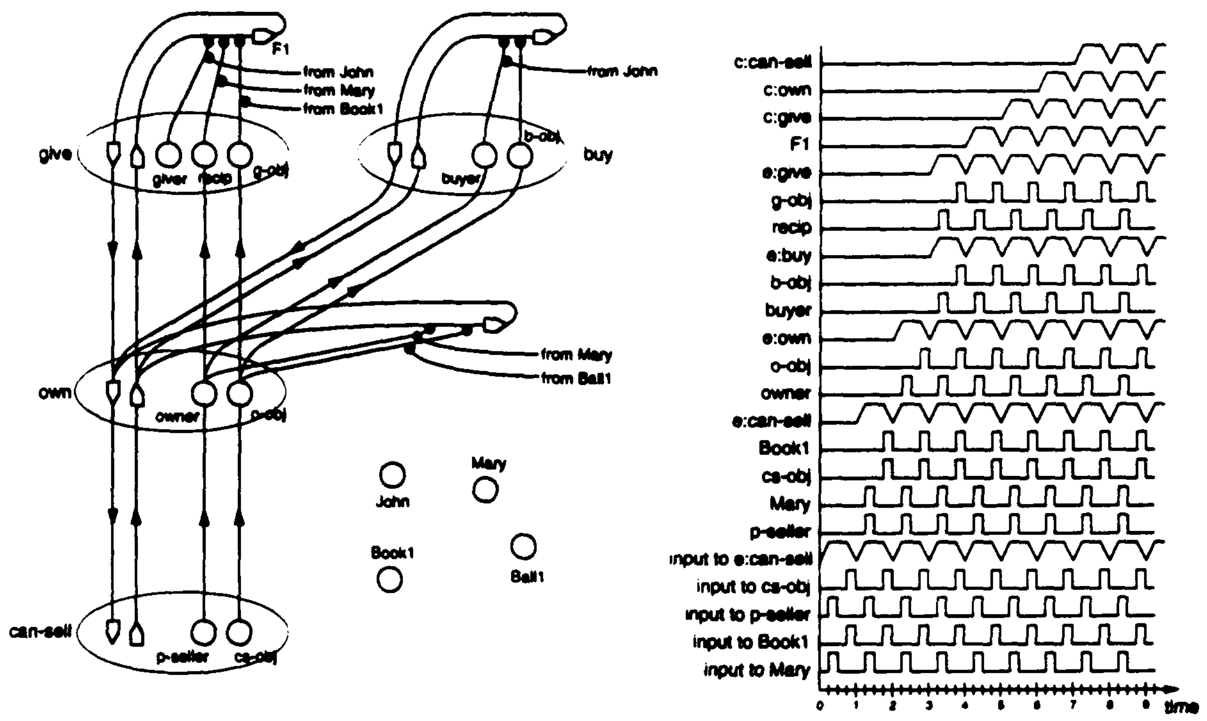


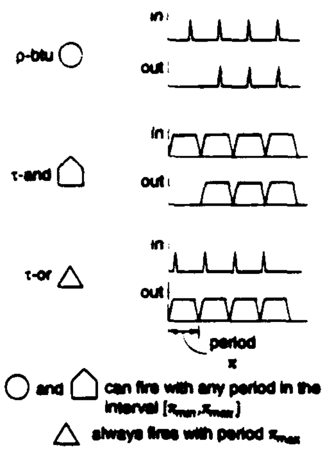
Figure 1: (a) An example encoding of rules and facts. (b) Activation trace for the query *can-sell(Mary,Book1)?*.

We briefly describe the reasoning system using an example. Figure 1a illustrates how long-term knowledge is encoded in the rule-based reasoning system. The network encodes the following *rules* and *facts*:

$\forall x,y,z$ [*give(x,y,z)* $\Rightarrow$ *own(y,z)*],
$\forall x,y$ [*buy(x,y)* $\Rightarrow$ *own(x,y)*],
$\forall x,y$ [*own(x,y)* $\Rightarrow$ *can-sell(x,y)*],
give(John,Mary,Book1),
buy(John,x), and
own(Mary,Ball1).

Rule and fact encoding makes use of several types of nodes (see Figure 2): **ρ-btu** nodes (depicted as circles), **τ-and** nodes (depicted as pentagons) and **τ-or** nodes (depicted as triangles). These nodes have the following idealized behavior: On receiving a spike train, a ρ-btu node produces a spike train that is *synchronous* (i.e., *in-phase*) with the driving input. We assume that ρ-btu nodes can respond in this manner as long as the inter-spike distance, π, lies in the interval $[\pi_{min}, \pi_{max}]$. Here π_{min} and π_{max} are the minimum and maximum inter-spike gaps for which the system can sustain synchronous activity (Shastri and Ajjanagadde, 1993). A τ-and node behaves like a *temporal* AND node, and becomes active on receiving an uninterrupted pulse train. On becoming active, a τ-and node produces a pulse train comparable to the input pulse train. A τ-or node on the other hand becomes active on receiving *any* activation; its output is a pulse whose width and period equal π_{max}. Figure 2 summarizes node behavior. The encoding also makes use of *inhibitory modifiers*—links that impinge upon and inhibit other links. A pulse propagating along an inhibitory modifier will block a pulse propagating along the link it impinges upon. In Figure 1a, inhibitory modifiers are shown as links ending in dark blobs.

Each entity in the domain is encoded by a ρ-btu node. An n-ary predicate P is encoded by a pair of τ-and nodes and n ρ-btu nodes, one for each of the n arguments. One of the τ-and nodes is referred to as the



ρ-btu
in
out
τ-and
in
out
τ-or
in
out
period
π
and can fire with any period in the interval [π_min, π_max]
always fires with period π_max

Figure 2: Behavior of ρ-btu, τ-and and τ-or nodes in the reasoning system.

enabler, *e:P*, and the other as the *collector*, *c:P*. In Figure 1a, *enablers* point upward while *collectors* point downward. The *enabler e:P* becomes active whenever the system is being queried about *P*. On the other hand, the system activates the *collector c:P* of a predicate *P* whenever the system wants to assert that the current dynamic bindings of the arguments of *P* follow from the knowledge encoded in the system. A rule is encoded by connecting the *collector* of the antecedent predicate to the *collector* of the consequent predicate, the *enabler* of the consequent predicate to the *enabler* of the antecedent predicate, and by connecting the arguments of the consequent predicate to the arguments of the antecedent predicate in accordance with the correspondence between these arguments specified in the rule. A fact is encoded using a τ-and node that receives an input from the enabler of the associated predicate. This input is modified by inhibitory modifiers from the argument nodes of the associated predicate. If an argument is bound to an entity in the fact then the modifier from such an argument node is in turn modified by an inhibitory modifier from the appropriate entity node. The output of the τ-and node is connected to the *collector* of the associated predicate. Figure 1a shows the encoding of the facts *give(John,Mary,Book1)* and *buy(John,x)*. The fact *give(John,Mary,Book1)* states that 'John gave Mary Book1' while *buy(John,x)* implies that 'John bought *something*'.

3.1 The Inference Process

Posing a query to the system involves specifying the query predicate and its argument bindings. The query predicate is specified by activating its *enabler* with a pulse train of width and periodicity π. Argument bindings are specified by activating each entity, and the argument nodes bound to that entity, in a distinct *phase*, phases being non-overlapping time intervals within a period of oscillation.

We illustrate the reasoning process with the help of an example. Consider the query *can-sell(Mary, Book1)?* (i.e., Can Mary sell Book1?) The query is posed by (i) Activating the *enabler e:can-sell*; (ii) Activating *Mary* and *p-seller* in the same phase (say, ρ_1), and (iii) Activating *Book1* and *cs-obj* in some other phase (say, ρ_2). As a result of these inputs, *Mary* and *p-seller* fire synchronously in phase ρ_1 of every period of oscillation, while *Book1* and *cs-obj* fire synchronously in phase ρ_2. See Figure 1b. The activation from the *can-sell* predicate propagates to the *own*, *give* and *buy* predicates via the links encoding the rules. Eventually, as shown in Figure 1b, *Mary*, *p-seller*, *owner*, *buyer* and *recip* will all be active in phase ρ_1, while *Book1*, *cs-obj*, *o-obj*, *g-obj* and *b-obj* would be active in phase ρ_2. The activation of *e:can-sell* causes the enablers of all other predicates to go active. In effect, the system is asking itself three more queries—*own(Mary,Book1)?*, *give(x,Mary,Book1)?* (i.e., Did *someone* give Mary Book1?), and *buy(Mary,Book1)?*. The τ-and node F1, associated with the fact *give(John,Mary,Book1)* becomes active as a result of the uninterrupted activation it receives from *e:give*, thereby answering *give(x,Mary,Book1)?* affirmatively. The activation from F1 spreads downward to *c:give*, *c:own* and *c:can-sell*. Activation of *c:can-sell* signals an affirmative answer to the original query *can-sell(Mary,Book1)?*.

3.2 The Type Hierarchy

Integrating a type hierarchy with the reasoning system (Mani and Shastri, 1993) allows the use of types (categories) as well as instances in rules, facts, and queries. This has the following consequences:

- The reasoning system can combine rule-based reasoning with *inheritance* and *classification*. For example, such a system can infer that 'Tweety is scared of Sylvester', based on the generic fact 'cats prey on birds', the rule 'if x preys on y then y is scared of x' and the *is-a* relations 'Sylvester is a cat' and 'Tweety is a bird'.
- The integrated system can use category information to *qualify* rules by specifying restrictions on the type of argument fillers. An example of such a rule is:

  $\forall x{:}\textit{animate},\ y{:}\textit{solid-obj}\ [\ \textit{walk-into}(x,y) \Rightarrow \textit{hurt}(x)\ ]$

  which specifies that the rule is applicable only if the two arguments of 'walk-into' are of the type 'animate' and 'solid-object', respectively.

Each entity (concept or instance) is now represented as a cluster of nodes and is associated with two *type-hierarchy switches*—a *top-down* T-switch and a *bottom-up* T-switch. Any entity can now accommodate up to k_1 dynamic instantiations, k_1 being the multiple instantiation constant for concepts. The T-switches regulate the flow of activation so as to ensure efficient and automatic dynamic allocations of instantiations.

3.3 Multiple Dynamic Instantiation of Predicates

Extending the reasoning system to incorporate multiple instantiation of predicates (Mani and Shastri, 1993) provides SHRUTI with the ability to *simultaneously* represent multiple dynamic facts about a predicate. For example, the dynamic facts *loves(John,Mary)* and *loves(Mary,Tom)* can now be represented *at the same time*. As a result, we can represent and reason using a set of rules which cause a predicate to be instantiated more than once. We can now encode rules like:

$\forall x,y\ [\ \textit{sibling}(x,y) \Rightarrow \textit{sibling}(y,x)\ ]$ and
$\forall x,y,z\ [\ \textit{greater-than}(x,y) \land \textit{greater-than}(y,z) \Rightarrow \textit{greater-than}(x,z)\ ]$

thereby introducing the capability to handle limited symmetry, transitivity and recursion.

Introduction of multiple dynamic instantiation of predicates relies on the assumption that, during an episode of reflexive reasoning, any given predicate need only be instantiated a *bounded* number of times. In (Shastri and Ajjanagadde, 1993), it is argued that a reasonable value for this bound is around three. We shall refer to this bound as the multiple instantiation constant for predicates, k_2.[2]

Predicate representations are augmented so that each predicate can represent up to k_2 dynamic instantiations. Each predicate also has an associated *multiple instantiation switch* (or M-switch) through which all inputs to the predicate nodes are routed. The switch arbitrates the input and brings about efficient and automatic dynamic allocation of predicate instantiations.

[2]This is the factor that limits symmetry, transitivity and recursion, since each predicate can accommodate at most k_2 dynamic instantiations.

4 Mapping SHRUTI onto Massively Parallel Machines

When mapping SHRUTI onto any massively parallel machine, several issues need to be taken into consideration in order to obtain effective performance and to strike a compromise between resource usage and response time. Several of these issues are discussed here. The discussion here is applicable when mapping SHRUTI onto any massively parallel machine. Later sections bring out how these issues are resolved in actual implementations on the CM-2 and CM-5. We have chosen the CM-2 and CM-5 as our target machines since they are representatives of their class and offer similar user interfaces and program development environments.

4.1 Exploiting Constraints Imposed by SHRUTI

As brought out in the previous sections, SHRUTI is a *limited* inference system, and imposes several psychologically and/or biologically motivated constraints in order to make reasoning tractable:

- The number of distinct entities that can participate in an episode of reasoning is bounded.
- Entities and predicates can only represent a limited number of dynamic instantiations.
- The form of rules and facts that can be encoded is constrained.
- The depth of inference is bounded.

The motivation for these constraints and their impact are discussed in (Shastri and Ajjanagadde, 1993). In terms of mapping SHRUTI onto parallel machines, it would be to our advantage to exploit these constraints to the fullest extent to obtain efficiency, speed and rapid response with large knowledge bases. Of course, if any of these constraints can be relaxed without paying a severe performance penalty, we would like to obtain a more powerful system by relaxing these constraints.

4.2 Granularity

For effective mapping, the SHRUTI network encoding a knowledge base must be partitioned among the processors in the machine. The network partitioning can be specified at different levels of granularity. At the fine-grained *network-level*, the partitioning would be at the level of the basic nodes and links constituting the network. A more coarse-grained *knowledge-level* mapping would partition the network at the level of knowledge elements like predicates, concepts, facts, rules and *is-a* relations.

The appropriate level of granularity for a given situation depends on several factors including the characteristics of the network, the processing power of individual processors on the machine and interprocessor communication mechanisms.

4.3 Network-Level Mapping

At this fine-grained level of granularity, the network is viewed as a collection of *nodes* and *links*. Factors that need to be taken into consideration when using network-level partitioning include:

Processor Allocation Nodes and links in the network should be assigned to processors on the target machine so as to minimize response time. Several options are possible: Each node and link could be assigned to a separate processor; groups of nodes and/or links could be assigned to a single processor; processors could be partitioned so that some handle only nodes and some handle only links; and so on.

Nodes The network which SHRUTI uses to encode a knowledge base consists of several different types of nodes. A given processor could handle only one type of node or could simulate an assorted combination of node types. The complexity of the node function should also be taken into consideration.

Links Like nodes, the links can also be of several types—including weighted, unweighted and inhibitory links. Placement of the links (on processors) relative to the placement of the nodes they connect is important since this is a major factor determining the volume of interprocessor communication.

Communication and Computation The partitioning scheme used to assign network components to processing elements should take into account the balance of computation and communication in the resulting system. Communication between network nodes, and hence interprocessor communication, is an essential aspect of connectionist network simulation. Trying to eliminate or unduly minimize interprocessor communication could lead to severe load imbalances whereby a few of the processing elements are overburdened with computation. Trying to evenly spread the computational load among the processing elements could result in increased communication and poor performance. A well designed system should strike a compromise between communication and computation so as to achieve effective performance.

4.4 Knowledge-Level Mapping

Knowledge-level mapping views the network at a relatively abstract level. At this granularity, knowledge base elements like predicates, concepts, facts, rules and *is-a* relations form the primitives. As is evident from Section 3, each primitive is constituted by a group of nodes and/or links. The behavior of these primitives is directly simulated without recourse to the underlying nodes and links constituting the primitive. Issues at this level include:

Predicates Each predicate could be assigned to a separate processor, or a group of predicates could be assigned to a single processor. In the latter case, predicates constituting a rule could all be placed on the same processor or could be scattered on different processors. Grouping predicates on any given processor could reduce the number of messages required to spread activation, but would make load balancing more difficult.

Facts Facts could be stored on the same processors to which the corresponding fact predicates have been assigned. An alternative approach would be to have dedicated processors for encoding facts. Such processors will receive inputs from both the fact predicate and the type hierarchy, and will signal fact matches globally or by communicating with the processor containing the predicate under consideration. In any case, we may need some mechanism to circumvent the situation where processors run out of memory since predicates could have a large number of associated facts.

Concepts Concept clusters are used in the type hierarchy to represent types and instances. Apart for being linked up to form the type hierarchy, these clusters must also communicate with the rule base. Careful choice of the mechanisms used to communicate the firing phase of concepts to the rule base could make the system more effective and reduce the number of messages exchanged in the system.

Rules When encoding rules, effective placement of predicates can minimize communication costs. The arbitration mechanism for accommodating multiple instantiations of a predicate also needs to be taken into account.

- When encoding rules, there are several choices available for the placement of predicates constituting the rule:
  - Depending on the processor allocation scheme used, we could allocate predicates occurring in a rule to the same processor. This would reduce interprocessor communication since fewer messages are required when the rule fires. This may not be easy to accomplish if predicates present in the rule being encoded have already been assigned to different processors.
  - A weaker form of the above scheme is to allocate predicates in a rule locally—i.e., on nearby processors. This scheme is easier to execute but will require relatively more messages in order to fire a rule.

  - The other extreme is to scatter the predicates randomly. Though this would require more messages, and messages would travel an average distance longer than for the previous two schemes, there are indications that random allocation may distribute messages uniformly over the entire machine instead of localizing it to "hot spots" where all the action happens, and would therefore reduce the incidence of message collisions (Leighton, 1992). Further, this scheme would provide better load balancing when answering a query.

- Identifying suitable performance measures and attempting to optimize these will aid in the objective placement of predicates when encoding rules. The performance measure could take into account factors like load balancing, cost of computation and communication, etc. It should be easy to compute the measure—or at least approximate it—using only local information.
- Predicate instance arbitration mechanisms ("switches") may need to be redesigned. When one or more predicates are assigned to each processor, switches may be unnecessary. Space ("banks") can be allocated for k_2 instances of each predicate. Incoming activation can be received in a buffer and then allocated to an empty bank under program control.

Type Hierarchy Most of the issues raised above will also need to be reconsidered with respect to the location and interaction of concepts in the type hierarchy. We would also need to streamline the interaction between the type hierarchy and the rule base for enhanced efficiency and effectiveness. Extending the scheme mentioned above for dealing with multiple instantiation, we might be able to do away with the type hierarchy T-switch.

It should be evident that most of the concerns addressed above are intertwined in that choosing one aspect will affect the choice of other aspects of the mapping. On a global scale, our aim is to develop an efficient and effective mapping by ensuring load balancing, minimizing interprocessor communication and by efficiently using resources including processors and memory.

Further, we believe that knowledge-level partitioning is the appropriate granularity for both the CM-2 and CM-5. The processing elements on the CM-2 are reasonably powerful (Appendix A) while the processing elements on the CM-5 (Section 6) are full-fledged SPARC processors. Thus, subnetworks corresponding to knowledge-level primitives can be implemented using appropriate data structures and associated procedures without necessarily mimicking the detailed behavior of individual nodes and links in the subnetwork.

5 SHRUTI on the CM-2

Initially we developed SHRUTI-CM2, a data parallel implementation of SHRUTI on the Connection Machine CM-2 (TMC, 1991a). A detailed description of SHRUTI-CM2, including design, knowledge encoding, spreading activation and performance characteristics can be found in Appendix A. However, due to the overwhelmingly superior performance of the SPMD implementation on the CM-5 (Section 6), the SHRUTI-CM2project was abandoned. Figure 3 compares the performance of the CM-2 and CM-5 parallel rapid reasoning systems.

6 SHRUTI on the CM-5

The Connection Machine model CM-5 (TMC, 1991b) is an MIMD machine consisting of anywhere from 32 to 1024 powerful processors.[3] Each processing node is a general-purpose computer which can execute instructions autonomously and perform interprocessor communication. Each processor can have up to 32 megabytes of local memory[4] and optional vector processing hardware. The processors constitute the leaves of a *fat tree* interconnection network, where the bandwidth increases as one approaches the root of the tree.

[3] In principle, the CM-5 architecture can support up to 16K processors.

[4] The amount of local memory is based on 4-Mbit DRAM technology and will increase as DRAM densities increase.

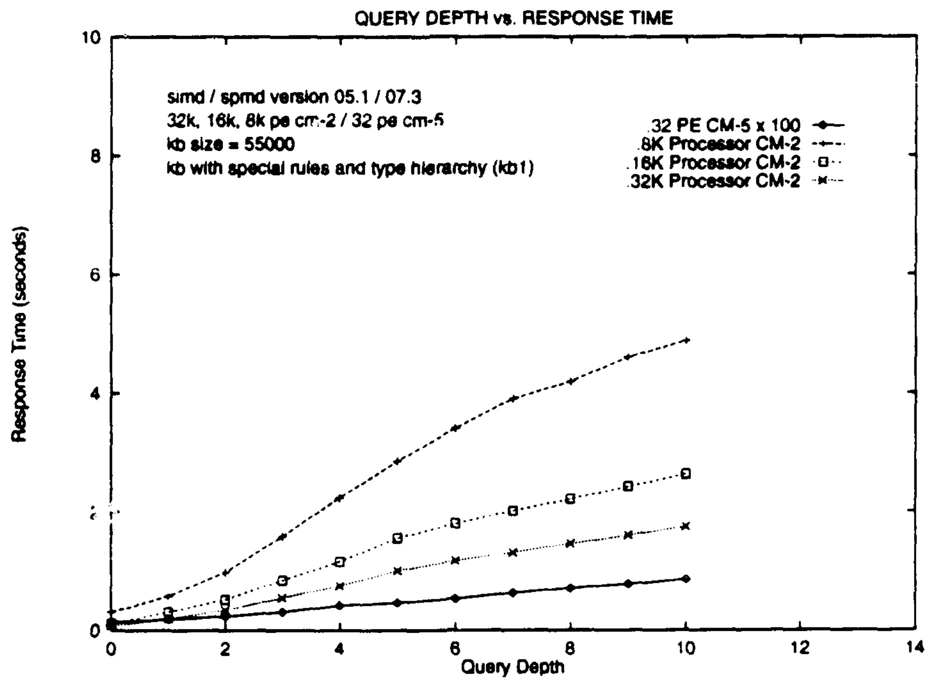


Figure 3: A comparison of SHRUTI-CM2 running on 32K, 16K and 8K processor CM-2 machines and SHRUTI-CM5 running on a 32 PE CM-5. The same full-fledged, structured, random knowledge base with special rules and a type hierarchy was used on all the machines. Note that the timing curve for the CM-5 has been multiplied by 100. Queries used were not randomly generated.

Every CM-5 system may have one or more control processors which are similar to the processing nodes but are specialized to perform managerial and diagnostic functions. A low-latency control network provides tightly coupled communications including synchronization, broadcasting, global reduction and scan operations. A high bandwidth data network provides loosely coupled interprocessor communication. A standard network interface connects nodes and I/O units to the control and data networks. The virtual machine emerging from a combination of the hardware and operating system consists of a control processor acting as a partition manager, a set of processing nodes, facilities for interprocessor communication and a UNIX-like programming interface. A typical user task consists of a process running on the partition manager and a process running on each of the processing nodes.

Though the basic architecture of the CM-5 supports MIMD style programming, operating system and other software constraints restrict users to SPMD (Single Program Multiple Data) style programs (TMC, 1994). In SPMD operation, a single program runs on all the processors, each acting on its share of data items. Both data parallel (SIMD) and message-passing programming on the CM-5 use the SPMD model. If the user program takes a primarily global view of the system—with a global address space and a single thread of control—and processors run in synchrony, the operation is data parallel; if the program enforces a local, node-level view of the system and processors function asynchronously, the machine is used in a more MIMD fashion. We shall consistently use "SPMD" to be synonymous with the latter mode of operation. In this mode, all communication, synchronization and data layout are under the programs' explicit control.

In this section we describe the design and implementation of the SPMD asynchronous message passing parallel rapid reasoning system—SHRUTI-CM5—that has been developed for the CM-5.[5]

[5]SHRUTI-CM2, the SIMD parallel rapid reasoning system for the CM-2, can also be run on the CM-5. Results of these experiments are described in Appendix C

6.1 Design Considerations

Granularity of Mapping

The individual processing elements on the CM-5 are powerful processors and therefore a subnetwork in the connectionist model can be implemented on a processor using appropriate data structures and associated procedures without necessarily mimicking the detailed behavior of individual nodes and links in the subnetwork. This entails that knowledge-level partitioning (Section 4) is the appropriate granularity for mapping SHRUTI onto the CM-5.

Active Messages and Communication

SHRUTI-CM5 uses CMMD library functions (TMC, 1993) for broadcasting and synchronization, while almost all interprocessor communication is achieved using CMAML (CM Active Message Library) routines.

CMAML provides efficient, low-latency interprocessor communication for short messages (TMC, 1993; Eicken et al., 1992). Active messages are asynchronous (non-blocking) and have very low communication overhead. A processor can send off an active message and continue processing without having to wait for the message to be delivered to its destination. When the message arrives at the destination, a handler procedure is automatically invoked to process the message. The use of active messages improves communication performance by about an order of magnitude compared with the usual send/receive protocol. The main restriction on such messages is their size—they can only carry 16 bytes of information. However, given the constraints on the number of entities involved in dynamic bindings (≈ 10), there is an excellent match between the size of an active message and the amount of variable binding information that needs to be communicated between predicate instances during reasoning as indicated by SHRUTI. SHRUTI-CM5 exploits this match to the fullest extent.

6.2 Encoding the Knowledge Base

In the SHRUTI-CM5 system, the knowledge base is encoded by presenting rules and facts expressed in a human readable, first-order logic-like syntax specified in Appendix D. The commands recognized by SHRUTI-CM5 are described in Appendix E.

Input Processing

Knowledge encoding in SHRUTI-CM5 is a two-part process:

1. **Serial preprocessing.** A serial preprocessor running on a workstation processes the input knowledge base and partitions it into as many chunks as there are processors on the CM-5 partition. The preprocessor outputs a set of files which are subsequently read by the CM-5 in parallel.
2. **Parallel knowledge base encoding.** Each processor on the CM-5 independently and asynchronously encodes the fragment of the knowledge structure assigned to it by the preprocessor. Depending on the processor assignment scheme used, each processor on a n processor CM-5 would typically need to process only $\frac{1}{n}$-th of the entire input knowledge base.

This two-part, asynchronous parallel input processing is well suited for large-scale knowledge bases. In addition, SHRUTI-CM5 also provides a direct input mode. In this mode, the processors cooperatively and synchronously encode the knowledge base. This mode can be used to by-pass serial preprocessing and is useful when small knowledge base fragments need to be added to an existing (large) knowledge base. SHRUTI-CM5 also supports convenient and consistent parallel updating of large knowledge bases via incremental preprocessing.

```
typedef struct cm_predbank          /* predicate bank on the CM */
{
  /* no fields used to encode KB */

  byte   collector;
  byte   enabler;
  byte   args[MAX_ARGS];            /* arg activation phase */
  char   qDepth;                    /* depth of reasoning chain
                                       which makes c: active */
} CM_PredBank;

typedef struct cm_pred              /* predicate on the CM */
{
  byte                noOfArgs;
  struct cm_list      *rules;       /* list of rules with pred as conseq */
  struct cm_list      *facts;       /* list of facts for pred */

  byte                nextFree;     /* index of next free bank (minst) */
  struct cm_predbank bank[K2];      /* predicate banks */
  struct cm_list      *ruleBPtr[K2]; /* rule back-pointers (for c: activation) */
} CM_Pred;
```

Figure 4: Structures used to represent predicates in SHRUTI-CM5. **MAX-ARGS** is the maximum number of arguments a predicate can have. **K2** is the multiple instantiation constant for predicates. The top part of the **typedefs** contain fields used to encode the knowledge base while the bottom part has fields used in a given episode of reasoning.

In either of the input modes, the knowledge base is scanned by a lexical analyzer and parser. Parsing the input results in the construction of internal data structures which represent the input presented to the system. A specially designated *server* processor builds hash tables which keep track of processor assignments. Whenever the system needs to know which processor houses some predicate P, the server broadcasts the required information. The system is designed in such a manner that the server does not become a bottleneck during the reasoning process. Information from the server is needed only when posing a query.[6] Once a query has been posed, the system data structures are so configured that spreading activation will proceed without the need for any information from the server. Maintaining a server processor therefore does not affect inference timing in any way.

Once a rule or fact (including an *is-a* relation) has been recognized and processed, the resulting internal data structures will be used to encode the rule or fact on the Connection Machine processors. In the case of a query, the data structures will be used to pose the query to the system.

Representing Knowledge Base Elements

Knowledge base elements—predicates, concepts, rules, facts and *is-a* relations—are assigned to a processor where the knowledge base elements are represented using suitable structures. Any knowledge base element is allocated space on exactly one processor. Figures 4–7 define the structures used to encode knowledge base elements. All processors in the partition except the server can encode knowledge base elements. The SHRUTI network is internally encoded by a series of pointers which serve to link predicate and concept representations.

[6]The server is also accessed when encoding knowledge in synchronous direct input mode.

```
typedef struct cm_rule                   /* rule slot on the CM */
{
  /* knowledge base encoding */
  struct cm_antlist  *antecedent;        /* list of antecedent predicates */
  struct cm_list     *consequent;        /* consequent predicate */
  byte               noOfAnts;           /* number of ant predicates for rule */
  int                weight;             /* weight; currently unused */
  byte               splCond[MAX_ARGS]; /* list of special conditions */
  int                splIndex[MAX_ARGS];/* procs containing spl cond constants */
  index              splPtr[MAX_ARGS];  /* ptr to spl cond constants */

  /* reasoning episode */
  byte   conseqCollector[K2];         /* c: values for the conseq pred are
                                         accumulated here; reqd for supporting
                                         multiple antecedent rules */
  char   qDepth[K2];                  /* reasoning chain depth; reqd for multiple
                                         antecedent rules */
} CM_Rule;

typedef struct cm_fact                         /* fact on the CM */
{
  struct cm_pred   *factPred;                  /* fact predicate */
  index            constant[MAX_ARGS];         /* fact argument pointers */
  index            constLocation[MAX_ARGS];   /* proc containing const */

  bool             active;                     /* fact active if set */
} CM_Fact;
```

Figure 5: Structures used to encode rules and facts in SHRUTI-CM5. MAX-ARGS is the maximum number of arguments a predicate can have. K2 is the multiple instantiation constant for predicates. Processor indices have type index and flags have type bool. Pointers are also of type index and index into local translation tables on the respective processors. The top part of the typedefs contain fields used to encode the knowledge base while the bottom part has fields used in a given episode of reasoning.

```
typedef struct cm_entitybank        /* entity bank on the CM */
{
  /* no fields used to encode KB */

  bool   buRelay;                   /* bottom-up relay */
  bool   tdRelay;                   /* top-down relay */
  byte   activation;                /* entity activation phase */
} CM_EntityBank;


typedef struct cm_entity                      /* entity on the CM */
{
  struct cm_list       *superConcepts;  /* bottom-up links */
  struct cm_list       *subConcepts;    /* top-down links */

  byte                 nextFree;        /* index of next free bank */
  struct cm_entitybank bank[K1];        /* entity banks */
} CM_Entity;
```

Figure 6: Structures used to represent entities in the type hierarchy (in SHRUTI-CM5). `K1` is the multiple instantiation constant for concepts in the type hierarchy. Flags have type `bool`. The top part of the **typedefs** contain fields used to encode the knowledge base while the bottom part has fields used in a given episode of reasoning.

```
typedef struct cm_isalink        /* is-a links on the CM */
{
  index           destination; /* index of destination proc */
  index           concept;     /* destination concept */

  /* no fields used during reasoning episode */
} CM_isALink;
```

Figure 7: Structure used to encode *is-a* relationships in SHRUTI-CM5. Processor indices have type `index`. Pointers are also of type `index` and index into local translation tables on the respective processors. The top part of the **typedef** contains fields used to encode the knowledge base while the bottom part has fields used in a given episode of reasoning.

```
initialize global statistics collection variables;

while (termination condition not met) {
  /* propagate activation in the type hierarchy */
  spread bottom-up activation;
  spread top-down activation;

  /* propagate activation in the rule base */
  back-propagate collector activation;
  check fact matches;
  propagate rule activation;

  update statistics collection variables;
}
```

Figure 8: The main propagation loop used in spreading activation during an episode of reasoning. The termination condition is met when the query is answered or the system determines that the query has no answer. Note that the order of the operations is crucial while propagating rule base activation. Activation of predicates whose collectors became active in the *previous* step must be back-propagated before facts are matched, since fact matching could activate other predicate collectors whose activation should be spread in the *next* propagation step. Further, fact matching for predicates that became active in the previous step must occur before new rules are fired, since firing rules could activate more predicates and fact matches for these predicates should be checked in the next iteration.

Unlike a serial machine, a "pointer" on the CM-5 would need both a memory address and the index of the processor to which the required fragment of memory belongs. In order to support parallel knowledge base encoding, the "memory addresses" are indirect and index into translation tables on the respective processors.

Encoding Rules and Facts

Depending on the processor allocation scheme (Section 4), every predicate and concept appearing in the knowledge base will be assigned to a processing node on the CM-5. Further, a rule or fact (including an *is-a* relation) that is being encoded will also be assigned to a processor. The actual details of the processor allocation are dictated by the processor assignment scheme being used. The SHRUTI-CM5 design offers several options for processor assignment schemes. SHRUTI-CM5 implementations use random processor assignment for predicates and concepts. Facts and *is-a* links are encoded on the processors containing the relevant predicate or concept[7] and rules were encoded on the processor containing the consequent predicate. Any processor in the machine (except the server) can have both predicates and concepts assigned to it.

Once the predicates, concepts and other knowledge base elements under consideration are assigned to processing elements on the CM-5, the knowledge base structures are built and/or updated. Rules, facts, and *is-a* links are encoded by a series of pointers which link predicate and concept representations to form the entire network.

[7]Assigning facts (*is-a* links) to the processor housing the associated predicate (concept) could result in deteriorating performance if the distribution of facts (*is-a* relations) is skewed—i.e., a few predicates (concepts) have a disproportionately large number of facts (*is-a* relations). Under such situations, other schemes such as splitting facts (*is-a* links) across processors may have to be considered.

6.3 Spreading Activation and Inference

Queries can be posed after the knowledge base has been encoded. Queries result in the activation of the relevant predicate and concepts as described in (Shastri and Ajjanagadde, 1993) and (Mani and Shastri, 1993). The activation propagation loop is shown in Figure 8. SHRUTI phases are represented as "markers"—integers with values ranging from 1 to the maximum number of phases.

The system runs asynchronously in that each processor continues with its processing irrespective of the progress made by other processors. If an answer to the query is found, the reasoning episode terminates immediately. If no answer is found after a certain number of asynchronous iterations, all processors synchronize and iterate synchronously. This synchronization ensures that activation has had a chance to traverse the depth of the network and is a safeguard against unlikely, but possible, cases of pathological imbalances in computation and interprocessor communication load. If no answer is found even after a fixed number of synchronous propagation steps, the reasoning episode terminates without an answer. This termination criteria is in keeping with the constraint that reflexive reasoning can only occur up to a bounded depth. The user can experiment with the terminating criteria by setting the number of asynchronous and synchronous iterations at compile time.

Each processing node (except the server), maintains several activation "frontiers" for both the rule base and the type hierarchy. Each frontier is essentially a list of predicates or concepts that are active and which need to be considered in the current activation propagation step. The following frontiers are maintained: A rule-frontier consisting of consequent predicates of rules under consideration in the current step; A fact-frontier consisting of predicates for which fact matches need to be checked; A back-propagation-frontier for handling back propagation of collector activation; and a type-hierarchy-frontier for activation propagation in the type hierarchy. During each propagation step, all frontiers are consistently updated in preparation for the next step in the iteration. Frontier elements are deleted after performing the required operation. A frontier element will reappear in the frontier for the *next* propagation step only if the operation attempted in the current step was unsuccessful. This ensures that the same operation—like firing a specific rule, matching a fact or firing an *is-a* fact—is not unnecessarily repeated. All frontiers are created and deleted asynchronously on *each* processor.

During an episode of reasoning, all interprocessor communication—including firing rules, spreading activation in the type hierarchy and back-propagating collector activation—is effected using active messages supported by the CMAML routines. The system has been tailored so that any information that needs to be exchanged between two processors will always fit in a single active message.

Each activation propagation step (on a given processor) results in advancing all activation frontiers by one level. In a given propagation step, each processor scans its frontiers and takes appropriate action—like firing rules, matching facts, etc. The active messages these processors send out will invoke handlers when they arrive at their destination. The handler functions perform the requested action—like receiving an instantiation, updating relevant frontiers, and so on. In the asynchronous phase, each processing node operates independently of the others.

Type Hierarchy and Multiple Instantiation

The type hierarchy is handled in a manner that is essentially similar to the rule base. Spreading bottom-up and top-down activation is separate and sequential. As entities go active, they broadcast their activations to all the processors in the partition. The processors cache this information for fast, local access during fact matching and special condition checking. In order to handle multiple instantiation (also see Appendix B), whenever a predicate or concept receives activation, it is compared with existing activation in the banks. If the incoming activation is not already represented, it is then deposited into the next available bank. The predicate representing the instantiation keeps track of the source of the instantiation in order to back propagate collector activation. An instantiation will need to be identified using (i) the processor housing the predicate or concept; (ii) the predicate or concept that originated the instantiation and (iii) the bank under consideration.

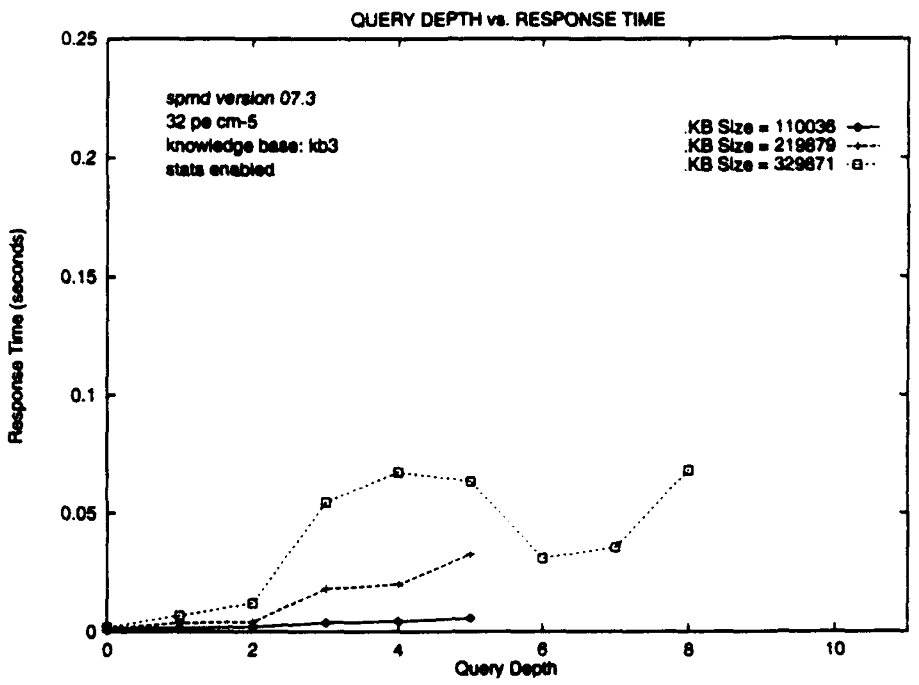


Figure 9: SHRUTI-CM5 running on a CM-5 with 32 processors. The graph shows the effect of the size of the knowledge base on response time for queries with varying inference depths. Due to the random nature of the knowledge base and the queries used, response times for a given depth are statistically reliable only when a large number of data points are averaged. For the larger depths, very few data points were available and this accounts for the seemingly better performance at larger depths. We expect the "dip" in the curve to "straighten out" as more data points are averaged.

Enough information is maintained when an instantiation is received so that collector activation can be propagated back to the predicate bank which originated the activation. Note that multiple instantiation is handled without the use of switches (Mani and Shastri, 1993); the above protocol is functionally equivalent to these switches and ensures that (i) any predicate or concept represents at most a bounded number of instantiations (the number being decided by the multiple instantiation constants K1 and K2) and (ii) a given instantiation is represented at most once so that no two banks of a predicate or concept represent the same instantiation.

Statistics Collection

SHRUTI-CM5 can be configured to collect statistics about various aspects of the system like knowledge base parameters, processor communication and computation, and the reasoning process. These include the distribution of knowledge base items among processors, the processor load and message traffic during query answering, and a count of knowledge base items of each type (rules, facts, concepts, etc.) activated during processing. Full-fledged data collection can slow down the system due to the extra time needed to accumulate required data.

6.4 Characteristics of SHRUTI-CM5

SHRUTI-CM5 has been tested using knowledge bases containing up to several hundred thousand rules and facts. Most of the experimentation has been carried out on a 32 node machine.

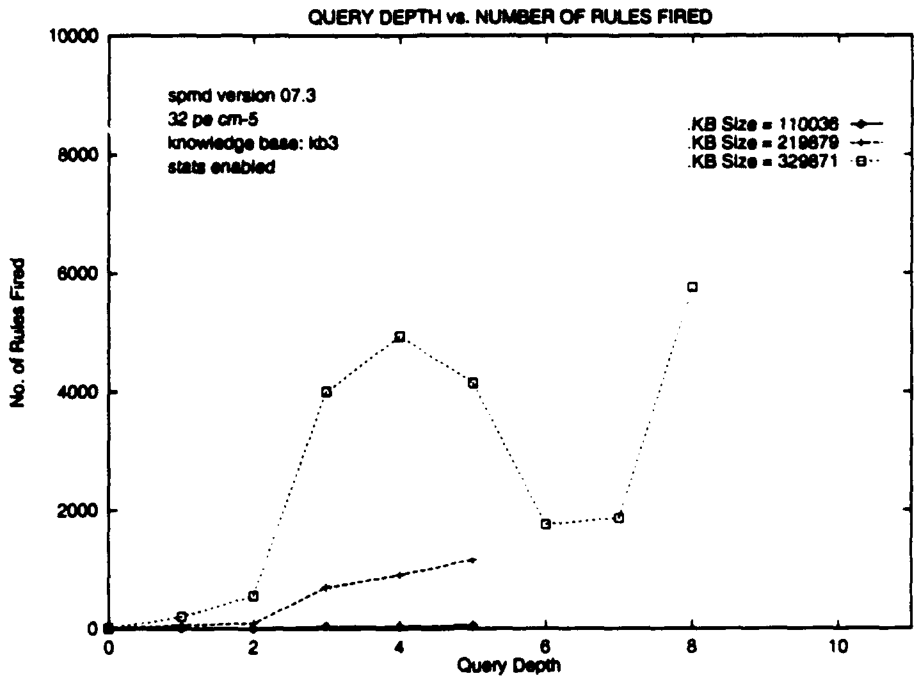


Figure 10: SHRUTI-CM5 running on a CM-5 with 32 processors. The graph shows the number of rules fired in answering queries with varying inference depths. See caption for previous figure for an explanation of the unexpected "dip" in the curve.

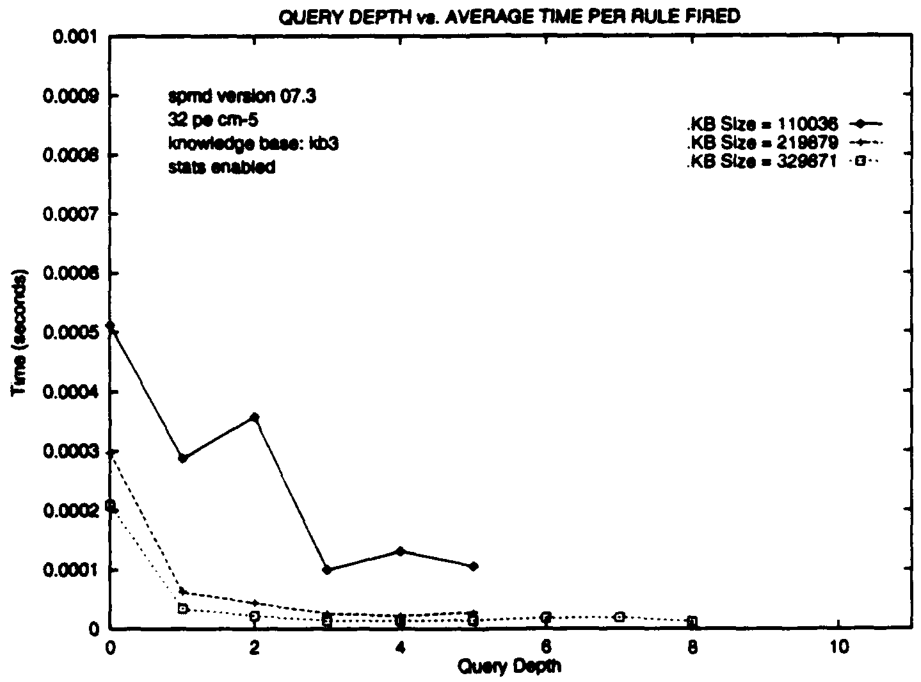
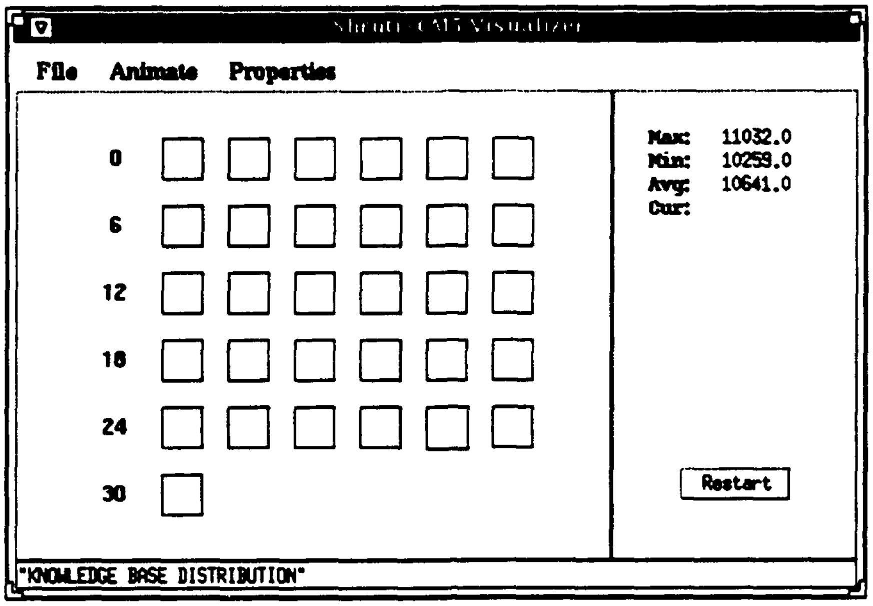


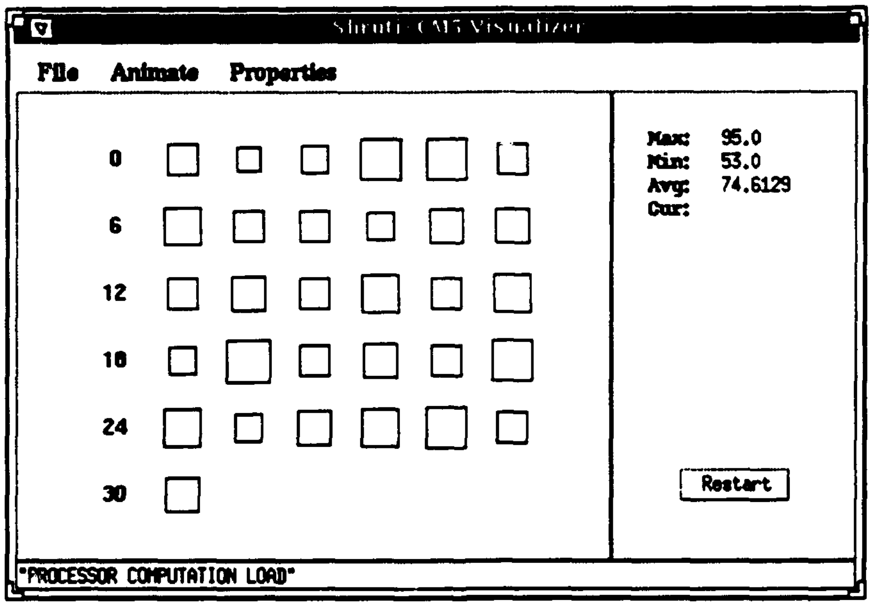
Figure 11: SHRUTI-CM5 running on a CM-5 with 32 processors. The graph shows the average time needed to fire a rule, shown as a function of knowledge base size and query depth.



Max: 11032.0
Min: 10259.0
Avg: 10641.0
Cur:

"KNOWLEDGE BASE DISTRIBUTION"

Figure 12: SHRUTI-CM5 running on a CM-5 with 32 processors. Distribution of knowledge base elements (rules, facts and *is-a* relations) on the CM-5 processors for a knowledge base with approximately 300,000 elements.



Max: 95.0
Min: 53.0
Avg: 74.6129
Cur:

"PROCESSOR COMPUTATION LOAD"

Figure 13: SHRUTI-CM5 running on a CM-5 with 32 processors. Computational load distribution on the CM-5 processors. The number of active predicates, entities and facts on each processor is shown. This load distribution was obtained when answering a query of depth 8 with a knowledge base of size approximately 300,000.

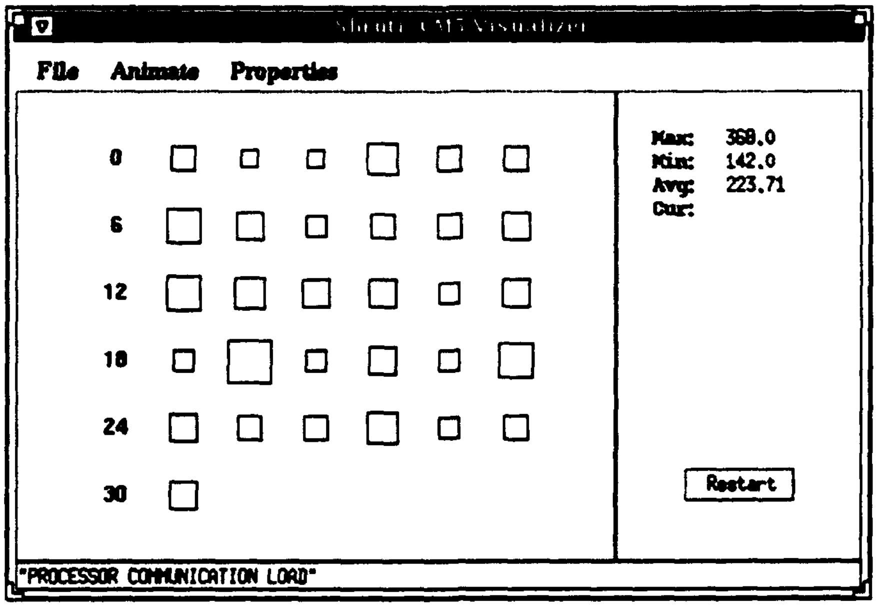


Figure 14: SHRUTI-CM5 running on a CM-5 with 32 processors. Communication load distribution on the CM-5 processors. The number of active messages sent by each processor is shown. This load distribution was obtained when answering a query of depth 8 with a knowledge base of size approximately 300,000.

Figures 9—14 illustrate the performance, timing and resource usage of SHRUTI-CM5. Figure 9 plots response time for varying query depths and knowledge base sizes. Figure 10 shows the number of rules fired when answering the respective queries. In both these figures, the queries used were generated randomly, and the values shown are averages for a given knowledge base size and query depth. About 100 queries with depths ranging from 0 to 8 were used; some of the queries were answered while several were not. The graphs depict the average for queries that were answered. The number of queries contributing to each data point ranges from about 15 (for depth 0) to 1 (for maximum depth). As the number of queries averaged over increases, we expect the curves to get smoother and statistically more reliable.

Figure 11 shows the average time needed to fire a rule as a function of knowledge base size and query depth. When a reasonably large number of rules fire in a given reasoning episode, the time needed per rule firing settles to a small, relatively constant value. Due to random queries being posed to a random knowledge base, there is lot of variation in the response time and other performance statistics for a given knowledge base size and query depth. Among all this variation, the behavior of the "time-per-rule" metric seems to be consistent over a variety of knowledge bases. We however do not know whether the "time-per-rule" metric will remain constant if the knowledge bases are significantly larger than the ones we have experimented with.

Figure 12 shows the distribution of a knowledge base with approximately 300,000 elements among the CM-5 processors. It is easily seen that the distribution is very even as a result of random processor allocation. Finally, Figures 13 and 14 show the computation and communication load on each processor for a 300,000 element knowledge base and a query of depth 8. Computation load is measured as the number of active predicates, entities and facts on each processor, while communication load is the number of active messages sent out by each processor. In spite of the unpredictable nature of the activation trail in the knowledge base, communication and computation load are relatively well balanced. Processor load is reasonably balanced irrespective of the query.

The timing reported in the graphs is the elapsed time needed to process the queries. Random, structured knowledge bases were used in these tests (see Section 6.5). These knowledge bases exploited the full functionality of the reasoning system and had a mix of regular rules and facts, rules with special conditions, quantified facts and *is-a* relations. Rules with special conditions included rules with repeated variables, typed variables, existential variables and entities; rules with multiple predicates in the antecedent and rules which lead to multiple instantiation of predicates. In spite of the large scale of these experiments, it is evident that SHRUTI-CM5 provides relatively good performance. Figure 3 compares the performance of SHRUTI-CM5 and SHRUTI-CM2.

6.5 Generating Knowledge Bases

Almost all experimentation with SHRUTI-CM5 have been carried out using randomly generated structured knowledge bases. Though the individual knowledge base elements are generated at random, these elements are organized into *domains* thereby imposing structure on the knowledge base. Each domain is a cluster of predicates along with their associated rules and facts. Domains could be of two types: *target* domains, which correspond to "expert" knowledge about various real-world domains; and *special* domains, which represent basic cognitive and perceptual knowledge about the world. A typical structured knowledge base would consist of several target domains and a small number of special domains. The predicates within each (target or special) domain, and predicates across target and special domains richly connected by rules; predicates across different target domains are sparsely connected. The structure imposed on the knowledge base is a gross attempt to mimic a plausible structuring of real-world knowledge bases. This is motivated by the notion that knowledge about complex domains are learned and grounded in metaphorical mappings from certain basic, perceptually grounded domains (Lakoff and Johnson, 1980). However, the "knowledge" in each domain is currently being generated at random.

The knowledge base generator takes several parameters as input. These parameters decide the number of predicates, entities, rules and facts that will be generated, the fractions of various special rules, facts and *is-a* relations, the number of domains, the distribution of the knowledge base among the domains and the fraction of inter- and intra-domain rules. The number and maximum depth of the type hierarchies generated can also be controlled.

The parameters supplied to generate the knowledge base used for the CM-5 experiments (identified in the graphs as **kb3**) is shown below:

```
------------ Knowledge Base Parameters ------------
 Number of rules:                       150000
 Number of facts:                       150000
 Number of predicates:                  50000
 Number of concepts:                    50000
 Multiple antecedent rule fraction:     0.10
 Multiple instantiation rule fraction:  0.10
 Special rule fraction:                 0.40
 Fraction of is-a facts:                0.25
 Fraction of facts with E vars:         0.10
------------ Domain Parameters --------------------
 Number of special domains:             3
 Number of target domains:              150
 Spl-Tgt knowledge base split:          0.02
 Fraction of intra-special-domain rules: 1.00
 Fraction of inter-special-domain rules: 0.00
 Fraction of intra-target-domain rules: 0.96
 Fraction of inter-target-domain rules: 0.01
 Number of type hierarchies:            10
 Maximum depth of type hierarchies:     5
 Fraction of shared leaves in type hiers.: 0.05
```

6.6 Proposed Experiments with Real-World Knowledge Bases

Recently we have obtained WordNet (Miller et al., 1990) and plan to map it to our system. Although WordNet does not exercise the full expressive and inferential power of our system, it is a sufficiently large knowledge structure with numerous applications and can be used to test the effectiveness of certain aspects of our system design, especially those having to do with message passing. We have also obtained a large knowledge base consisting of over 14,000 frames and 170,000 attribute-value pairs about plant anatomy and physiology from Bruce Porter of the University of Texas at Austin (Porter et al., 1988). The mapping of this knowledge base to our system is very similar to that of WordNet. We are also trying to acquire a subset of the CYC knowledge base (Lenat et al., 1990) in the near future.

A planned application of our knowledge base system is to couple it to the Berkeley Restaurant Project (BeRP) speech understanding system being developed at teh International Computer Science Institute (Jurafsky et al., 1994a; Jurafsky et al., 1994b). BeRP functions as a knowledge consultant whose domain is restaurants in the city of Berkeley, California. Users ask spoken language questions of BeRP which then queries a database of restaurants and gives advice based on cost, type of food, and location. The current BeRP system cannot perform inferences and any possible inferences are either hard wired into the grammar or added to the restaurant database. Our knowledge base system will allow BeRP to make inheritance-like inferences (a Chinese restaurant is an Asian restaurant) as well as more complex inference (if the user has a car they can get to more distant restaurants). The rapid response of our knowledge base system will be particularly useful for an on-line speech understanding system like BeRP.

6.7 The SHRUTI-CM5 User Interface

The following example illustrates the existing user interface to SHRUTI-CM5 and supporting utilities.

1. **Knowledge base generation.** The user must begin with a knowledge base in a syntax recognized by SHRUTI-CM5. Knowledge bases in other formats should be translated into a from accepted by the system. The following is an example knowledge base in SHRUTI-CM5 syntax.

```
/* Rules */
Forall x,y,z  [ give(x,y,z) => own(y,z) ];
Forall x,y    [ own(x,y) => can_sell(x,y) ];
Forall x:Animal, y:Animal
              [ preys_on(x,y) => scared_of(y,x) ];
Forall x,y,z Exists t
              [ move(x,y,z) => present(x,z,t) ];
Forall x,y,z  [ move(x,y,z) => present(x,y,t) ];
Forall x,y    [ sibling(x,y) & born_together(x,y) => twins(x,y) ];
Forall x,y    [ sibling(x,y) => sibling(y,x) ];

/* Facts */
give (John,Mary,Book1);
move (John,Nyc,Boston);
sibling (John,x);
Forall x:Cat, y:Bird [ preys_on(x,y) ];
Exists x:Robin [ own(Mary,x) ];

/* Type hierarchy */
is-a (Bird,Animal);
is-a (Cat,Animal);
is-a (Canary,Bird);
```

```
is-a (Tweety,Canary);
is-a (Sylvester,Cat).
```

It is also possible to create a (pseudo-random) knowledge base using the knowledge base generator (Section 6.5). The output of the generator is in the above syntax.

2. **Preprocessing and loading.** The preprocessor reads the input knowledge base, assigns knowledge base items to CM-5 processors (using one of several available processor assignment schemes) and writes out a set of files. These files are read and encoded on the CM-5.

3. **Parallel knowledge processing.** Once the KB has been loaded on the CM-5 one can pose queries, obtain answers, and gather performance and timing data. The following dialog illustrates how the user interacts with the system. The system prompt is >>. User input is in `typewriter` font while system output is shown in *slanted* font.

```
>> i input-kb.pp
     Processing file input-kb.pp .... done
>> m -g
>> i
     Enter Rules/Facts or Query:
     can_sell(Mary,Book1)?
>> r
     Simulating ... done
     Query answered affirmatively in 0.001638 seconds
>> z
     Resetting network ... done.
>> i query
     Processing file query .... done
>> r
     Simulating ... done
>>
```

The input command `i` is used to input the knowledge base and to pose queries. The run command `r` runs a reasoning episode. It reports elapsed time if the query is answered (as in the case of `can_sell(Mary,Book1)?`). If the query is not answered, no timing is displayed (as in the case of the query contained in the file `query`). Further commands can be used to view knowledge base distribution on the processors, processor load, individual processor timing, number of rules fired, active predicates and concepts, number of messages sent, and so on (see Appendix E).

The system also provides the capability to process command files in order to facilitate unattended batch processing.

4. **Analysis and visualization.** The data obtained from reasoning episodes can be analyzed and plotted as graphs (Figures 9–11); dynamic processor load, timing, etc. can be visualized (Figures 13 and 14); knowledge base distribution can be analyzed and visualized (Figure 12); and the actual connectivity of the knowledge base can be graphically displayed. All analysis and visualization are done off-line.

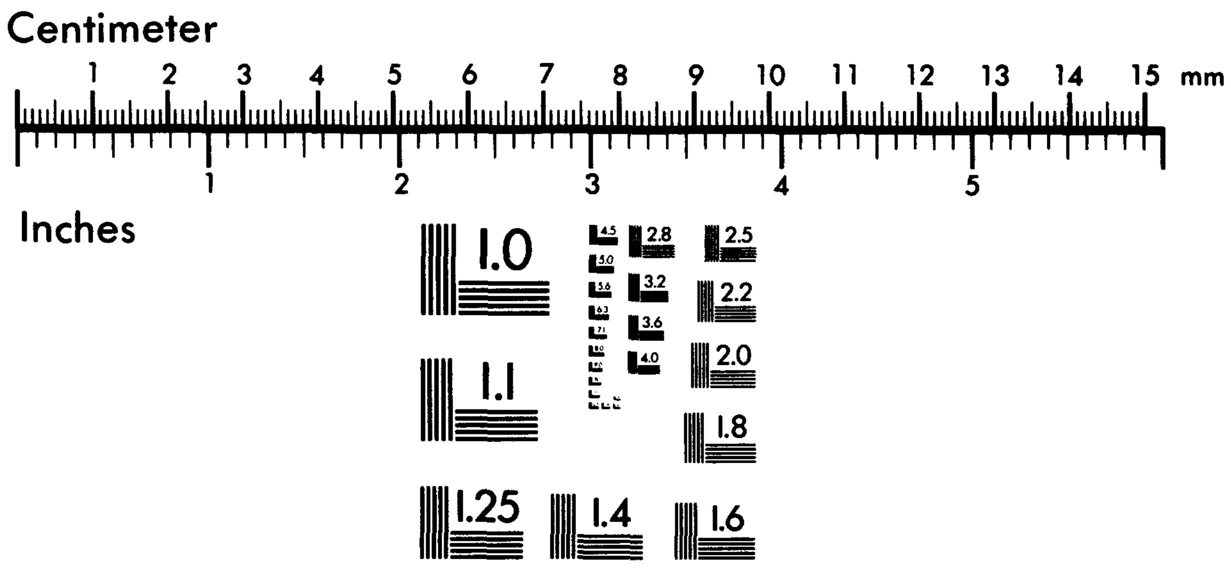
Integrated User Environment

In the existing SHRUTI-CM5 system, all tools and utilities are separate programs. The user must manually invoke the required program or script in order to execute any kind of processing, analysis or visualization. Future versions of SHRUTI-CM5 will provide an integrated graphical user environment which integrates a suite of programs and tools:

AD-A289 251 SPATIO-TEMPORAL NEURAL NETWORKS FOR VISION REASONING AND RAPID DECISION MAKING(U) INTERNATIONAL COMPUTER SCIENCE INST BERKELEY CA L SHASTRI 31 AUG 94 XB-ONR 3/3

UNCLASSIFIED N00014-93-1-1149 NL

END
FILMED
DTIC


Centimeter
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 mm
1 2 3 4 5
Inches
1.0 1.1 1.25 1.4 1.6 1.8 2.0 2.2 2.5 2.8 3.2 3.6 4.0 4.5 5.0 5.6 6.3 7.1 8.0

- Random knowledge base generators and programs for translating knowledge bases in (a limited number of) other formats into a form accepted by SHRUTI-CM5;
- A parser for reading knowledge expressed in a human readable, first-order-logic-like language;
- A preprocessor for transforming knowledge bases into a form suitable for efficient loading onto the underlying parallel machine;
- An efficient, rapid reasoning system running on the underlying parallel machine which can retrieve answers to queries in real-time.
- Statistics collection procedures which can accumulate data regarding the knowledge base and various aspects of parallel knowledge processing.
- A variety of tools for analyzing and visualizing the knowledge base and data produced by the statistics modules.

The parallel rapid reasoning system would form the core of the SHRUTI-CM5 system around which all the other programs and tools would be organized. Data processing, analysis and visualization tools would be a combination of scripts, already existing tools and custom generated programs. Except for the parallel part, all the other tools would be off-line, usable on a workstation and integrated into an interactive, easy-to-use graphical interface.

The SHRUTI-CM5 system would also provide for automated remote access to the CM-5 so that all off-line tools and processing can be confined to the local workstation. The parallel reasoning episodes will be run on the remote CM-5 and the results and output transferred back to the local workstation for further processing.

7 Related Work

There has been considerable work in the conceptual design of massively parallel systems based on spreading activation, marker passing, and connectionism (Lange and Dyer, 1989; Sun, 1991; Barnden and Srinivas, 1991; Waltz and Pollack, 1985; Charniak, 1983; Fahlman, 1979). However, only very few researchers have tried to implement knowledge base systems on existing parallel platforms. A salient example of such work is the PARKA system (Evett et al., 1993) implemented on the CM-2. PARKA encodes frame-based knowledge (analogous to a semantic network) and supports efficient computation of inheritance, recognition, and structure retrieval which is a generalization of recognition. The performance of PARKA has been tested using pseudo-random networks (with up to 130,000 nodes) as well as subsets of CYC (Evett et al., 1993; Lenat et al., 1990). The CYC subsets used had about 26,000 units. PARKA's run time for inheritance queries is $O(d)$ and for recognition queries is $O(d+p)$ where d is the depth of the *is-a* hierarchy and p is the number of property constraints. Actual run-times range from a fraction of a second (for inheritance queries) to a little more than a second (for recognition queries with 15–20 conjuncts). PARKA does not support rule-based reasoning; it can only handle frame-based knowledge with some extensions to deal with memory-based reasoning.

Semantic Networks on Special Purpose Hardware

Fahlman (1979) had proposed the design of NETL, a massively parallel machine that could execute marker passing algorithms for computing inheritance and recognition in parallel. Although this machine was never built, it influenced the design of the CM-2 (Hillis, 1985). Researchers such as Moldavan (1993) have also proposed and built special purpose hardware for realizing semantic networks and production systems.

The partitioning and mapping of production systems (or rule-based systems) onto multiprocessors is considered in (Moldovan, 1989). A performance index is obtained by analyzing rule interdependencies.

This performance index is optimized so as to maximize inherent parallelism and minimize interprocessor communication. Optimizing the performance index is intractable and approximations and simplifications are necessary in order to make the problem tractable. A message-passing multiprocessor architecture (RUBIC, for Rule-Based Inference Computer) for parallel execution of production systems is also described.

The Semantic Network Array Processor (SNAP) developed at the University of Southern California is described in (Moldovan et al., 1992). The conceptual design of the SNAP is based on associative memory and marker passing, and is optimized for representing and reasoning with semantic networks. The SNAP provides a special instruction set for network creation and maintenance, marker creation and propagation, logic operations and search/retrieval. A SNAP prototype has been built with off-the-shelf components and used to implement a parallel, memory-based parser (Moldovan et al., 1992). The parser is capable of processing sentences in 1–10 seconds depending on the sentence length and the size of the knowledge base used. The largest knowledge base used consisted of about 2,000 nodes.

Unlike SHRUTI and PARKA, SNAP-based knowledge representation systems use special purpose hardware. Further, SNAP-based systems can only deal with semantic networks and do not currently support the full range of inferences supported by SHRUTI.

8 Conclusion

We have described an SPMD mapping of SHRUTI on the Connection Machine CM-5. We have discussed issues involved in the design and implementation of this system—both from machine independent and machine dependent points of view. From the test results summarized in the previous sections, it is evident that SPMD implementations are vastly superior in comparison with the SIMD system and offer speedups of several hundred. In view of its greatly improved performance, we plan to expend our effort in improving and extending the asynchronous (SPMD) message passing system on the CM-5. The SPMD rapid reasoning system on the CM-5 is also being mathematically analyzed (Mani, 1994) with the objective of obtaining quantitative measures which can be used to further improve performance.

SHRUTI-CM5[8] currently supports only backward reasoning. Future work on the CM-5 will involve developing a forward reasoning system and an integration of the forward and backward reasoners.

All experiments reported here have used randomly generated knowledge bases. As noted in Section 6.6, we plan to encode large real-world knowledge bases on the system and interface it with applications. This will not only help us evaluate the parallel rapid reasoning systems more thoroughly, but will also result in practical and usable systems. Depending on the kind of knowledge bases used, we also expect this endeavor to provide insights into aspects of reflexive reasoning.

9 Acknowledgements

This work has been supported by ARO grants DAA29-84-9-0027 and DAAL03-89-C-0031 to the Army Research Center at the University of Pennsylvania, ONR grant N00014-93-1-1149 and NSF resource grant CCR930001N to Lokendra Shastri and National Science Foundation Infrastructure Grant CDA-8722788 to the University of California at Berkeley. All CM-5 and 32K, 16K and 8K CM-2 experiments were run on the Connection Machines at the National Center for Supercomputing Applications (NCSA), University of Illinois at Urbana-Champaign. All experiments on the 4K CM-2 were run at the University of Pennsylvania. The CM-5 workshop at NCSA—conceived by the MetaCenter Computational Science Institutes in Parallel Computing, sponsored by the NSF, and organized by the National Center for Supercomputing Applications and the San Diego Supercomputing Center—provided D. R. Mani with some computing resources and a great learning opportunity.

[8] And SHRUTI-CM2, see Appendix A.

Thanks to David Bailey, Susan Davidson, Jerry Feldman, Ben Gomes, Dan Jurafsky, Marilynn Livingston, Srini Narayanan, Chris Overton, Sanguthevar Rajasekaran, and David Waltz for several comments and suggestions which have contributed to this work. Thanks to Tom Fontaine and the CM-2 support staff at the University of Pennsylvania for their help in getting started with the CM-2 and C*, the consulting staff at NCSA and the University at California at Berkeley for their help with the CM-5.

A SHRUTI on the CM-2

The CM-2 (TMC, 1991a) is an SIMD data parallel computing machine which can be configured with up to 64K processing elements. Each processor has several kilobits of local memory and can execute arithmetic and logical instructions, calculate memory addresses, read and store information in memory and perform interprocessor communication. The processors are organized as an n-dimensional hypercube. The CM-2 is controlled by a standard serial front end processor (usually a VAX or SUN machine). A sequencer decodes commands from the front end and broadcasts them to the data processors, all of which then execute the same instruction simultaneously and synchronously. A NEWS grid provides fast communication between adjacent processors and a router network provides general interprocessor communication between any two processors.

The design and implementation of the SIMD parallel rapid reasoning system on the CM-2—SHRUTI-CM2—is based on knowledge-level partitioning (Section 4) of the underlying network generated by a knowledge base. We describe techniques used to encode the knowledge base and implement spreading activation when answering queries. We then explore the characteristics of the system by running a battery of tests. All discussion pertains only to backward reasoning.

A.1 Encoding the Knowledge Base

The knowledge base is encoded by presenting rules and facts (including *is-a* facts) to the SHRUTI-CM2 system. The input syntax for rules, facts, *is-a* relations and queries is specified in Appendix D. Appendix E gives a listing of commands recognized by SHRUTI-CM2.

Input Processing

A lexical analyzer and parser read the input, parse it and build internal data structures which represent the rules and/or facts presented to the system. All input processing is performed sequentially on the front-end.

As predicates and entities (or concepts) are recognized in the input, the parser builds hash tables which keep track of processor assignments. The hash tables can be used to efficiently access these predicates and entities while encoding rules and facts, posing queries and inspecting their state.

Once a rule or fact (including an *is-a* relation) has been recognized and processed, the resulting internal data structures can be used to encode the rule or fact on the Connection Machine processors. In the case of a query, the data structures will be used to pose the query to the system.

Representing Knowledge Base Elements

Knowledge base elements are represented on the processors using *parallel* structures. A parallel structure allocates space for the specified structure on *every* processor. Figures 15 and 16 indicate the structures used to encode predicates, rules and facts in the rule-base. The structures used to encode concepts and *is-a* relationships in the type hierarchy are similar (though simpler). Note that a parallel structure will be allocated for each knowledge base element: predicate, fact, rule, concept and *is-a* link. When the knowledge base grows and more space is needed, the size of the parallel structure is doubled. The virtual processor capability of the CM-2 ensures that each (physical) processor now houses two structures. This is transparent to the programmer and one can still assume that each processor houses one structure, with double the number of (virtual) processors in the machine. Using this scheme, the representation automatically scales with the size of the knowledge base. As the number of virtual processors increases, the system will run proportionately slower. The virtual processor mechanism therefore provides a simple, scalable and transparent way of trading off time for space.

```
typedef struct cm_pred              /* predicate on the CM */
{
  bool               used;          /* flag */
  byte               noOfArgs;

  byte               nextFree;      /* index of next free bank (minst) */
  struct cm_predbank bank[K2];      /* predicate banks */
} CM_Pred;


typedef struct cm_predbank          /* predicate bank on the CM */
{
  /* no fields used to encode KB */

  bool   cChange;                   /* collector value changed */
  bool   eChange;                   /* enabler value changed */
  byte   collector;
  byte   enabler;
  byte   args[MAX_ARGS];            /* arg activation phase */
} CM_PredBank;
```

Figure 15: Structures used to represent predicates in SHRUTI-CM2. MAX-ARGS is the maximum number of arguments a predicate can have. K2 is the multiple instantiation constant for predicates. Flags have type bool. The top part of the typedefs contain fields used to encode the knowledge base while the bottom part has fields used in a given episode of reasoning.

Encoding Rules and Facts

Depending on the processor allocation scheme used (Section 4), every predicate and entity appearing in the knowledge base will be assigned to a (virtual) processing element on the CM-2. Further, a rule or fact (including an *is-a* relation) that is being encoded will also be assigned to a (virtual) processor. These two processor allocations—one for the relevant predicates/entities and the other for the rule/fact under consideration—may or may not be independent. The actual details of the processor allocation are dictated by the processor assignment scheme being used.

The current and more recent versions of SHRUTI-CM2 use random processor assignment schemes for all knowledge base elements—predicates, concepts, facts, rules and *is-a* links. Earlier versions used random allocation for predicates and concepts; however, facts and *is-a* links were encoded on the processors containing the relevant predicate or concept and rules were encoded on the processor containing the consequent predicate.

Once the predicates, concepts and other knowledge base elements under consideration are assigned to processing elements on the CM-2, all that remains to be done in order to encode the rule/fact is to correctly fill out the various fields in the relevant structures. Encoding a fact involves the corresponding predicate and the entities filling the arguments of the predicate. Encoding a rule (*is-a* relation) involves two predicates (concepts) and a rule-slot (*is-a* link). If a rule has multiple predicates in the antecedent, the encoding is slightly more complex, as pictured in Figure 17.

```
typedef struct cm_rule              /* rule slot on the CM */
{
  /* knowledge base encoding */
  bool   used;                      /* flag */
  bool   dummy;                     /* rule slot is dummy if flag set */
  index  antecedent;                /* invalid for head rule slots */
  index  consequent;                /* points to head slot in a dummy */
  byte   noOfAnts;                  /* > 1 in a head rule slot */
  int    weight;
  byte   antNoOfArgs;               /* invalid for head rule slots */
  byte   argMap[MAX_ARGS];          /* arg mapping; invalid on head slot */
  byte   splCond[MAX_ARGS];         /* not used in dummy slots */
  int    splIndex[MAX_ARGS];        /* not used in dummy slots */

  /* reasoning episode */
  byte   dummyCollector[K2];        /* used only in dummy slots */
  bool   fire;                      /* rule can fire if set */
  bool   selected;                  /* instantiation selected if set */
  byte   nextBank;                  /* next conseq pred bank to consider */
  byte   bankSelected[K2];          /* rule back pointer */
  /* NOTE: bankSelected[i] == j if bank i in the ant pred has
     instantiation from bank j in the conseq pred;  valid only on
     non-head rule slots; in a head rule slot bankSelected[i] == i */
} CM_Rule;


typedef struct cm_fact              /* fact on the CM */
{
  bool   used;                      /* flag */
  index  factPred;                  /* fact predicate index */
  byte   noOfArgs;
  index  constant[MAX_ARGS];        /* fact arguments */

  bool   active;                    /* fact active if set */
} CM_Fact;
```

Figure 16: Structures used to encode rules and facts in SHRUTI-CM2. MAX-ARGS is the maximum number of arguments a predicate can have. K2 is the multiple instantiation constant for predicates. Flags have type bool while processor indices have type index. The top part of the typedefs contain fields used to encode the knowledge base while the bottom part has fields used in a given episode of reasoning.

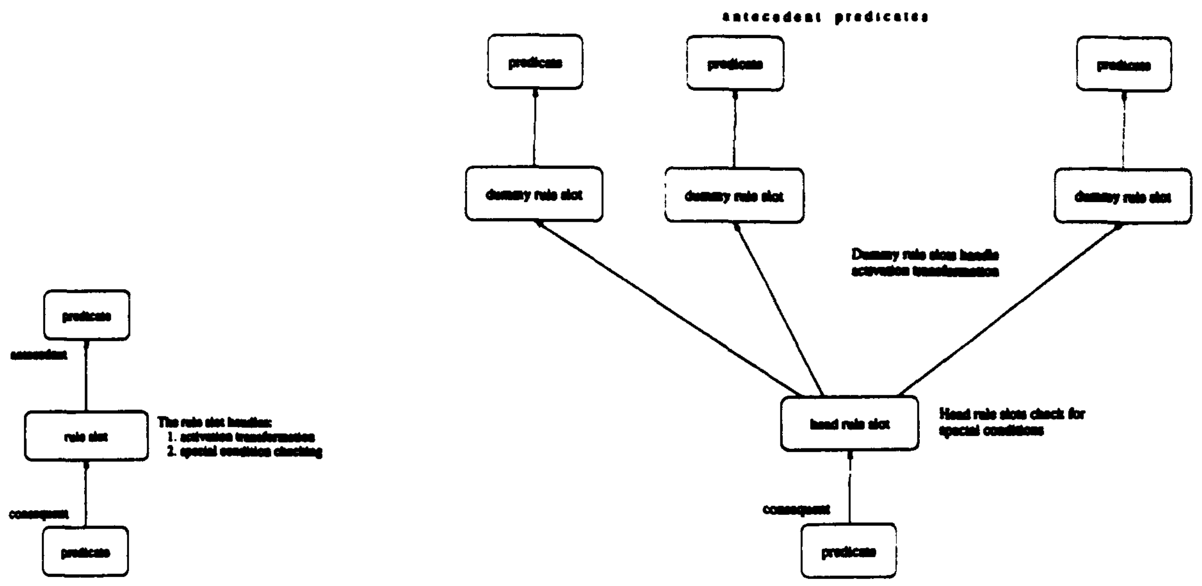


Figure 17: Encoding single- and multiple-antecedent rules. The figure on the left indicates the encoding of single-antecedent rules while the figure on the right depicts the encoding of multiple antecedent rules. Every predicate and rule-slot is housed on a processor. Arrows indicate links which are implemented using interprocessor communication.

A.2 Spreading Activation and Inference

Queries can be posed after the knowledge base has been encoded. Again, queries have a specific syntax (as described in Appendix D) and result in activating the relevant predicate and concepts in keeping with the description in (Shastri and Ajjanagadde, 1993) and (Mani and Shastri, 1993). The reasoning episode can then be run, either step-wise or to completion. We now describe the mechanics of spreading activation and matching facts in the system. The gross structure of the activation propagation loop is indicated in Figure 8. Phases in SHRUTI are represented as "markers"—integers with values ranging from 1 to the maximum number of phases.

The Rule Base

As shown in Figure 8 spreading activation in the rule base consists of three steps:

- *Propagating rule activation.* Spreading activation in the rule base by rule firing is achieved by executing the following:
  1. Every non-dummy rule-slot gets the instantiation in the consequent predicate bank under consideration.
  2. All non-dummy rule-slots check if all special conditions in the rule are satisfied.
  3. If all special conditions are satisfied, the dummy rule-slots get the respective instantiations from the corresponding head rule-slot.
  4. All non-head rule-slots transform the activation and send it to the respective antecedent predicates.

  In the process of firing a rule, the system maintains sufficient book-keeping information to back-propagate collector activation to the consequent of a rule.

Once a rule fires, it will not fire again unless a new bank of the consequent predicate becomes active. This ensures that the same rule does not repeatedly fire thereby minimizing unnecessary interprocessor communication. Note also that the processor housing the rule-slot will need to communicate with other processors in order to **get** predicate bank instantiations, **get** information from the head rule-slot, **send** information to dummy rule-slots and **send** the transformed activation to the antecedent predicate.

- *Checking fact matches for active predicates.* All facts for predicates which have active collectors are matched simultaneously. Processors encoding the facts communicate with the processors housing the relevant predicates and concepts in order to check if the firing "phases" match. If a fact "fires", the collector of the corresponding predicate is activated.
- *Back-propagating collector activation.* Sending collector activation to predicate banks which originated the activation involves the following:
  1. Non-head rule-slots **get** the state of the predicate collector.
  2. Dummy rule-slots send the collector value to the head rule slot which accumulates all the incoming values.
  3. Non-dummy rule-slots send the activation to the respective consequent predicates provided the collector activation exceeds a threshold. The threshold could depend on the number of antecedent predicates for the rule, the level of activation of antecedent predicate(s), and/or other factors.

  Rule-slots that have already propagated collector activation to the corresponding predicate bank will not participate in this step. Again, this is done in order to minimize unnecessary interprocessor communication.

The Type Hierarchy

Propagating activation in the type hierarchy is similar to spreading activation in the rule-base, except that it is much simpler. Spreading bottom-up activation and top-down activation are handled separately (and sequentially) in the type hierarchy. When spreading bottom-up (top-down) activation, all *is-a* links which have an active bank in the subconcept (superconcept) "fire" and spread activation to the respective superconcept (subconcept). The *is-a* link **gets** activation from the subconcept (superconcept) and **sends** it to the superconcept (subconcept). Again, in order to minimize communication, we ensure that any new activation traverses corresponding *is-a* links exactly once.

Multiple Instantiation

Multiple instantiation in SHRUTI-CM2 is handled without the use of switches (Mani and Shastri, 1993). Predicates and concepts can accommodate `K2` and `K1` instances respectively. When spreading activation in the network, predicate and concept banks are considered one at a time. In other words, in a given clock cycle (i.e., in one iteration of the propagation loop; see Figure 8) only one active bank of a predicate or concept will be considered. As described in Appendix B, care is taken to avoid potential problems that could result from this technique.

Whenever a predicate or concept receives activation, it is compared with existing activation in the banks. If the incoming activation is not already represented, it is then deposited into the next available bank. The rule- or link-slot that sent in the activation is notified that the instantiation it sent has been selected. In the rule base, the rule-slot receives the bank number accommodating the new instantiation. This information is needed when back-propagating collector activation. If the incoming activation is already represented in the predicate or concept, or if all banks are already in use, the incoming activation is discarded. Even in this case, rule-slots are notified so that they can proceed to the next bank of the consequent predicate. A

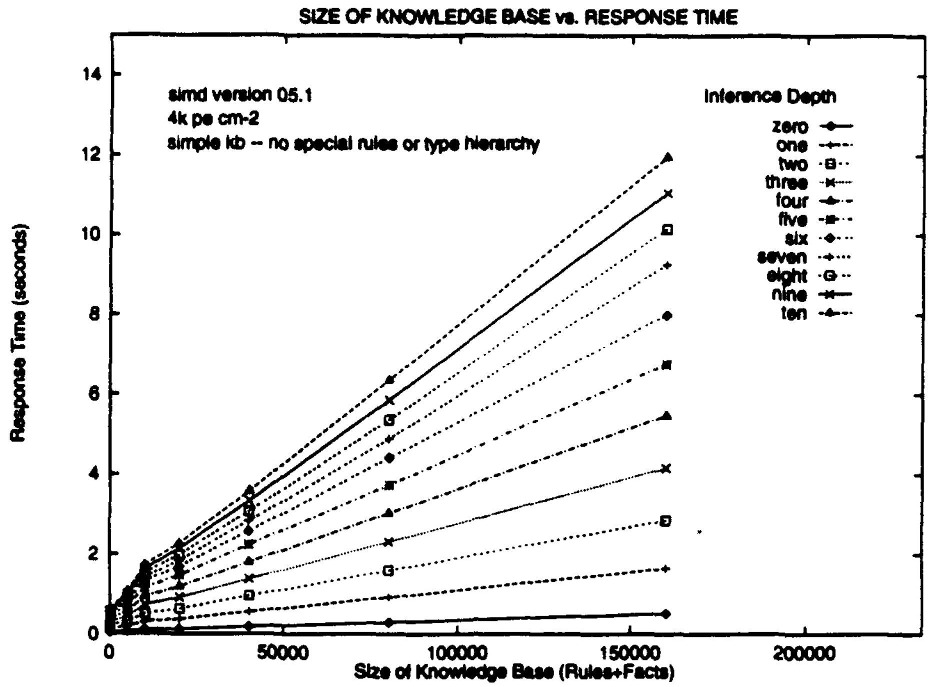


Figure 18: SHRUTI-CM2 running on a CM-2 with 4K processors. The graph shows the effect of the size of the knowledge base on response time for queries which require inference depths ranging from 0 to 10. Queries used were not randomly generated. The knowledge base used was not structured.

rule-slot retries sending the same instantiation if it does not receive notification that the activation was either selected or discarded.

The above protocol simulates the function of the multiple instantiation switches, and brings about efficient dynamic allocation of predicate and concept banks to ensure that:

- Any predicate (concept) in the system represents at most K2 (K1) instantiations.
- A given instantiation is represented at most once; in other words, no two banks represent the same instantiation.

Statistics Collection

Apart from timing the reasoning episodes, SHRUTI-CM2 can also be configured to gather data about several other aspects including knowledge base parameters (number of rules, facts, *is-a* relationships, and concepts) and communication data (number of messages, sends and gets). Enabling full-fledged data collection can slow down the system due to the extra time needed to accumulate the required data.

A.3 Characteristics of SHRUTI-CM2

SHRUTI-CM2 has been run on a 4K CM-2 and on a 32K CM-2. Both machines had 256 kilobits of memory on each processor. Figures 18 and 19 summarize the results of experiments run on these machines. In these figures, the response time shown is the actual CM time used. The timing routines available on the CM-2 also report elapsed time for the reasoning episode. Elapsed time is affected by other processes running on the front end and is therefore unreliable. The knowledge bases used in these experiments were generated

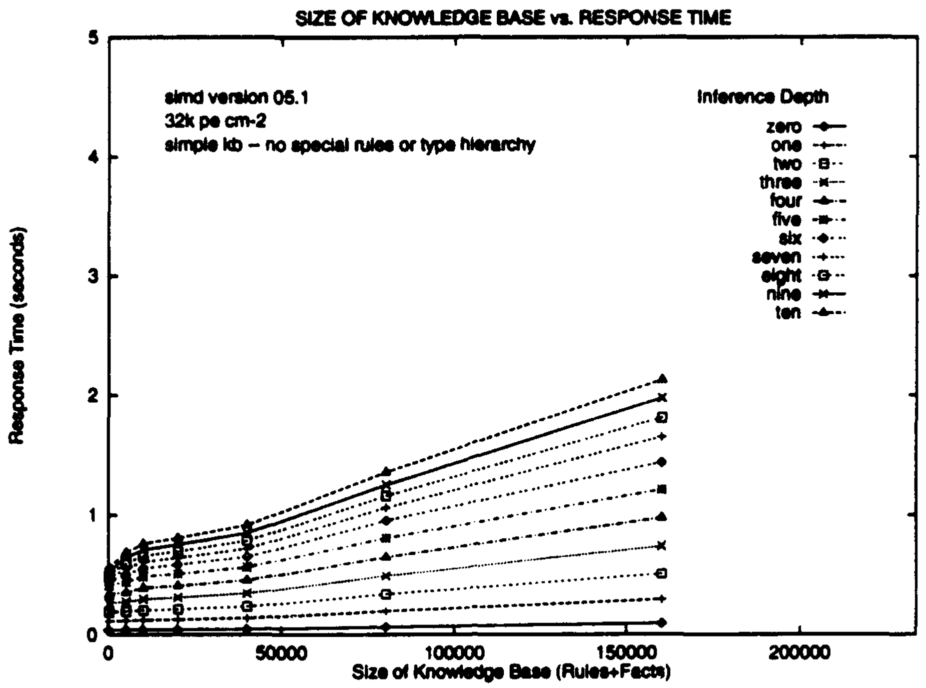


Figure 19: SHRUTI-CM2 running on a CM-2 with 32K processors. The graph shows the effect of the size of the knowledge base on response time for queries which require inference depths ranging from 0 to 10. Queries used were not randomly generated. The knowledge base used was not structured.

at random, and did not contain *is-a* relationships or rules with special conditions. The inference path for a given query was tailored to ensure a reasonable branching factor—at least one of the predicates in the activation frontier had five or more outgoing links originating from it.

Based on these and other experiments, and on the design of SHRUTI-CM2, we can summarize the characteristics of the system:

- The response time is approximately linear with respect to the size of the knowledge base, for knowledge bases with up to 160,000 elements. Thus, as the size of the knowledge base increased, query answering time increased proportionately. This is to be expected since more predicates would be active on the average and would entail proportionately more processing and interprocessor communication as the size of the knowledge base increases.

  Beyond a certain limit, we expect response time to increase steeply with the size of the knowledge base. However, effort was not expended in locating this limit or studying the characteristics of the system near this threshold since our focus shifted to the CM-5. As a result, all timing results stated here apply only to knowledge bases with up to 160,000 rules and facts.

- Time taken to answer a query increases as the average branching factor of the knowledge base increases. This again is caused by increased processing and interprocessor communication.

- Increasing inference depth needed to answer a query proportionately increases response time. Every extra inference step requires an extra activation propagation step (i.e., an extra iteration of the loop in Figure 8).

- Response time is approximately inversely proportional to the number of (physical) processing elements on the machine. This can be attributed to the increased computing power and the lower "density" (with fewer knowledge base elements per processor) which results in enhanced parallelism.

- The time taken to answer a query ranges from a fraction of a second to a few tens of seconds.
- An inherent problem with the use of parallel variables on the CM-2 is inefficient memory usage. Since the number of virtual processors must always be a power of two, this could potentially lead to significant waste of memory. There appears to be no simple solution to this problem without breaking out of SIMD operation. SPMD implementations on the CM-5 avoid this problem entirely.
- The maximum size of the knowledge base that can be encoded on a machine depends on the total amount of memory available on the machine. In addition, with increasingly large knowledge bases, the communication bottleneck would also significantly slow down the system.

B Multiple Instantiation—Some Technical Details

Multiple instantiation in both SHRUTI-CM2 and SHRUTI-CM5 is handled *without* the use of switches (Mani and Shastri, 1993). When spreading activation in the network, predicate and concept banks are considered one at a time. In other words, in a given iteration of the activation propagation loop (see Figure 8) only one active bank of a predicate or concept will be considered. This technique could cause indefinite waits in the rule base. To illustrate the problem, suppose we are currently considering bank i of predicate P. Let P be the consequent of rules r_1 and r_2. Let R_1 and R_2 be rule-structures that represent r_1 and r_2. At propagation step t, suppose r_1 fires and r_2 does not. The fact that r_1 fired for bank i of P will be noted in R_1, and R_1 can shift its focus to the next active bank $i+1$ in the next propagation step. Since r_2 did not fire, R_2 is stuck at bank i. R_2 cannot skip bank i and go on to bank $i+1$ since r_2 could fire later due to activation propagating in the type hierarchy. We circumvent this problem by defining special protocols.

Note that this problem does not arise in the type hierarchy since all *is-a* links originating at a concept always fire—unlike a rule, no preconditions need to be satisfied for an *is-a* link to fire.

SHRUTI-CM2

Let D_{th} be the depth of the type hierarchy. Then,

- If a rule r for bank i of some predicate fires at time step t, then update R, the structure representing r, to consider bank $i+1$ of the corresponding predicate in step $t+1$ (subject to the conditions mentioned below).
- If a rule r for bank i of some predicate does not fire at time step t, then two cases are possible:
  1. If $t \leq D_{th}$, then do not update R. Thus, bank i will be reconsidered in step $i+1$.
  2. If $t > D_{th}$, update R to consider bank $i+1$ in the next time step.[9]

Since activation spread in the type hierarchy will not activate any new concepts after D_{th} time steps, this scheme ensures that all banks of a predicate will eventually be considered.

SHRUTI-CM5

In SHRUTI-CM5, the multiple instantiation indefinite wait problem is handled by placing special elements on the rule-frontier. Normally, a rule-frontier element is a (consequent) predicate, along with the bank that was instantiated. All rules for that predicate bank are considered in a given propagation step. If any rule does not fire for this bank, then a special pair of elements is added to the rule-frontier. This pair specifies the

[9] Whenever any rule-slot R is updated to consider an inactive predicate bank, R waits till an instance has been assigned to that bank.

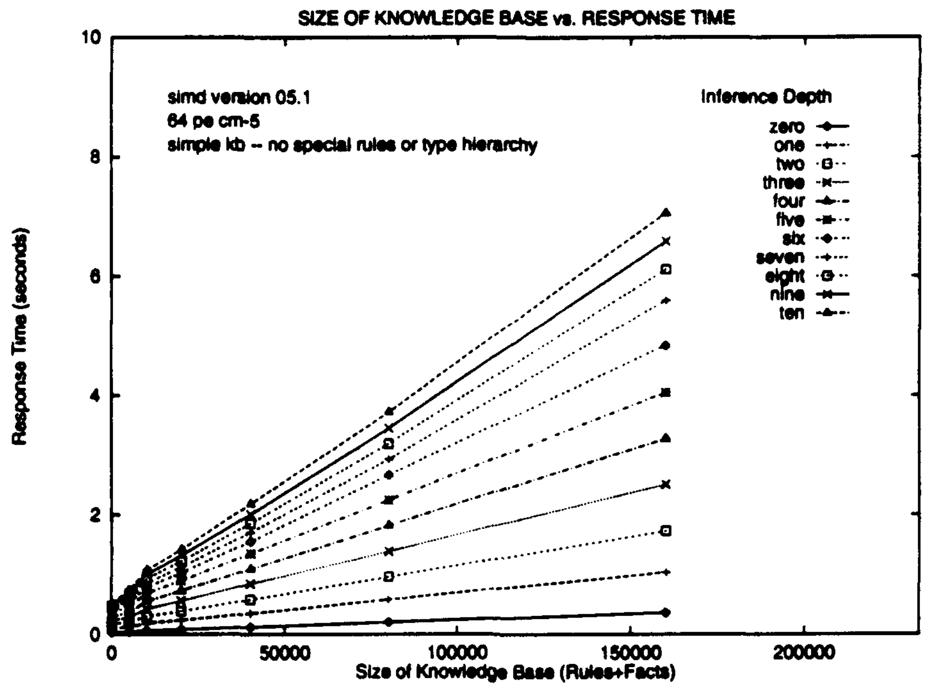


Figure 20: SHRUTI-CM2 running on a CM-5 with 64 processors. The processing nodes on the CM-5 are used in SIMD mode. The graph shows the effect of the size of the knowledge base on response time for queries which require inference depths ranging from 0 to 10. Queries were not randomly generated. The knowledge base used was not structured.

predicate bank *and* the associated rule that need to be reconsidered in the next propagation step. Whenever such a pair is encountered on the rule-frontier, only the specified rule is processed. If subsequent banks of the predicate become active, these predicate banks will be placed on the frontier as usual, irrespective of the fact that previous banks could have rules which have not yet fired.

C SHRUTI-CM2 on the CM-5

In this section, we briefly evaluate SHRUTI-CM2 running on the CM-5. Since SHRUTI-CM2 is written in C*, and a C* compiler is available for the CM-5, SHRUTI-CM2 was recompiled and run on the CM-5. SHRUTI-CM2 running on the CM-5 uses the CM-5 in data-parallel (SIMD) mode. Figure 20 summarizes the results. Comparing with Figures 18 and 19, we observe that the performance of SHRUTI-CM2 on the CM-5 is comparable to that on the CM-2[10], though message passing on the CM-5 appears to be more robust.

[10]The rule of thumb seems to be that a 32 node CM-5 is approximately equivalent to a CM-2 with 8K processing elements.

D Input Syntax for Rules, Facts and Queries

To illustrate the input syntax for rules, facts and *is-a* relations, we begin with an extension of the example in Section 6.7.

```
/* RULES */
forall x,y,z [give(x,y,z) => own(y,z)];
forall x,y   [buy(x,y) => own(x,y)];
forall x,y   [own(x,y) => can_sell(x,y)];
forall x,y   [sibling(x,y) & born_together(x,y) => twins(x,y)];
forall x,y   [preys_on(x,y) => scared_of(y,x)];
forall x,y,z [move(x,y,z) => present(x,z,t)];
forall x,y,z [move(x,y,z) => present(x,y,t)];
forall x,y exists t
             [born(x,y)    => present(x,y,t)];
forall x:Animate, y:Solid_obj
             [walk_into(x,y) => hurt(x)];

/* FACTS */
give (John, Mary, Book1);
give (x, Susan, Ball2);
forall x:Cat, y:Bird preys_on (x,y);
exists x:Robin [own(Mary,x)];

/* IS-A FACTS */
is-a (Bird,Animal);
is-a (Cat,Animal);
is-a (Robin,Bird);
is-a (Canary,Bird);
is-a (Tweety,Canary);
is-a (Sylvester,Cat).
```

NOTE: Any text included between /'s are comments. The comments given above are enclosed between /* ... */ so that they look identical to comments in C code.

The above example illustrates the input syntax accepted by the parallel rapid reasoning systems. Most of the features are self-evident. Some points to be noted regarding the input syntax follow. Items prefixed by a dagger (†) are supported only by SHRUTI-CM5.

- A rule meant for the backward reasoner is said to be *balanced* if the following conditions are satisfied:
  - Repeated variables in the antecedent are also present in the consequent.
  - Typed variables, existential variables and entities present in the antecedent are also present in the consequent.

  Only balanced rules will be accepted by the system. Rules which do not satisfy the above conditions will be rejected. A warning message to this effect will be printed.

- Any variable (used in a rule) which is not listed in either the list of universally quantified variables or in the list of existentially quantified variables is assumed to be existentially quantified.

- Any name beginning with an uppercase alphabetic character is assumed to be an entity. All names beginning with lowercase are variable names. Names of predicates can begin with either uppercase or lowercase letters. Capitalization of names should be consistently used — for example, **name1** and **Name1** would represent two *different* predicates; similarly, `Const_a` and `Const_A` are different entities.

- A semicolon (;) indicates that a rule, fact or *is-a* fact has been entered; it also indicates that more input is to follow. The occurrence of a period (.) in the input indicates the end of a rule, fact or *is-a* fact and also terminates the input. A (quantified or unquantified) predicate terminated by a ? is interpreted as a *query*.
- The lexical analyzer removes all whitespace; the input is therefore unaffected by the addition of extra blanks, tabs or newlines. Further, spaces can be omitted wherever it is not essential[11].
- The lexical analyzer also removes all *comments*. Any text enclosed between /'s (/ ... /) is a comment. The text of a comment can contain any character or symbol except /. A comment can start and end at any point in the input. In particular, a comment may span several lines or may be limited to part of a single input line.
- †**Tags.** Predicates and entities can be tagged (with a non-zero, positive integer) by using the < > construct: **<give(x,y,z),3>** or **<Mary,6>**. Tags can be used to group "similar" predicates and entities together.
- **Error Handling.** When syntax errors are detected in the input, the action taken depends on the mode of input:
  - If input is being read from the terminal (**stdin**), an error message is issued, and the *last* rule or fact should be re-entered after typing one or more semi-colons (;).
  - If input is being read from a file, the parser prints the line number containing the syntax error and continues reading the file, so that all syntax errors in the file are listed. Rules or facts in the input that were correctly recognized (i.e., had no syntax error) will be encoded; the others will be ignored.

Below is the formal grammar for the input language (for rules, facts, *is-a* relations and queries) which specifies the exact form of each input structure. The grammar is accurate for SHRUTI-CM5. Though most of the constructs are identical in SHRUTI-CM2, there are some minor differences. Further, SHRUTI-CM2 does not support tags.

```
input → .                               /* stop - no more input */
            | ; input                   /* continue - more input */
            | input-item input
input-item → query                      /* query */
            | fact                      /* fact */
            | rule                      /* rule */
            | tag-def                   /* tag definition */
rule → q-prefix [ pred-list => predicate ]
            | pred-list => predicate
fact → predicate
            | q-pred
query → predicate ?
            | q-pred ?
tag-def → < predicate , NUM >
            | < constant , NUM >
q-pred → q-prefix [ predicate ]
```

[11]To distinguish between the variable '**forallx**' and '**forall x**', a space is *essential.* But a space is not required after the ',' in '**own(x,y)**'. In general, spaces are not essential before and after punctuation symbols.

```
q-prefix → FORALL type-list
          | EXISTS type-list
          | FORALL type-list EXISTS type-list
          | EXISTS type-list FORALL type-list
type-list → variable
          | variable : constant
          | variable , type-list
          | variable : constant , type-list
pred-list → predicate & pred-list
          | predicate
predicate → arg-or-pred ( arg-list )
          | arg-or-pred ( )
arg-list → arg-or-pred , arg-list
          | arg-or-pred
arg-or-pred → constant | variable
constant → CONST
variable → VAR
```

Here, CONST represents entities (any token starting with an uppercase letter), VAR are variables (quantified or unquantified) in the rules, facts or queries and are tokens beginning with lowercase letters. The variable and entity tokens are represented by a sequence of alphanumeric characters along with _ and *. Any integer is recognized as a NUM. The tokens FORALL and EXISTS are recognized when the input contains these words, spelled with any combination of uppercase and lowercase letters (i.e., arbitrarily capitalized).

E SHRUTI-CM Commands

Commands recognized by SHRUTI-CM2 and SHRUTI-CM5 are listed below. Some of the commands and descriptions are applicable only to SHRUTI-CM5 and are prefixed by a dagger (†). The SHRUTI-CM5 preprocessor only supports the commands `i`, `w` and `q`. Each command is invoked by using a single character. The first non-blank character typed at the input prompt is taken to be the command. Any non-blank text following the first character forms the argument(s) for the command. The list below indicates the purpose of the command, the command syntax and a brief description of the command.

Quit Syntax: `q`
Terminates the SHRUTI-CM program.

Help Syntax: `?`
Prints out a list of available commands and the command-line options and/or arguments which the commands accept.

Read Input Syntax: `i [ -f | -b ] [input-file]`
Reads input from the terminal (when `input-file` is not specified) or a file (when `input-file` is specified). The `-b` option is used to build a backward reasoning system (default), while the `-f` option builds a forward reasoning system (currently unsupported).

†In SHRUTI-CM5 the behavior of this command is dictated by the current input mode. The system always starts up in parallel asynchronous mode; the mode can be changed using the `m` command. In parallel asynchronous mode, each processor in the partition processes a different input file `input-file.pid` where `pid` is a three digit processor index (prefixed by zeros if necessary). In global synchronous mode, all processors cooperatively process the same input file `input-file`.

†Syntax: `i [-h hash-table-file] [ -f | -b ] [input-file]`
The `-h` option for read input is supported by the SHRUTI-CM5 preprocessor and can be used to update the internal server hash tables which store processor assignment and other details for predicates and concepts. This feature is useful for incremental preprocessing of large knowledge bases.

†**Change Input Mode** Syntax: `m [ -p | -g ]`
Changes input mode to parallel asynchronous (with the `-p` option) or to serial, global synchronous (with the `-g` option). Without any option, this command prints out the current input mode. The current input mode dictates the behavior of the `i` command.

†**Write Out Hash Table** Syntax: `w [-o output-file-prefix]`
Writes out the current server hash tables to the specified file (with a `.hashtables` extension). If no output file prefix is given, `kb.pp` is used as default. The hash tables written out can be read by the preprocessor (using the `i` command with the `-h` option) and supports incremental preprocessing of large knowledge bases.

†Syntax: `w [ -g ] [-o output-file-prefix]`
This command, when used on the SHRUTI-CM5 preprocessor, writes out the preprocessed knowledge base. The output file names are suffixed with the processor number. If the output file prefix is not specified, `kb.pp` is used as the default. If the `-g` option is absent, the inference dependency graph for the knowledge base is also written out (with file extension `.idg`)

Run Reasoning Episode Syntax: `r [[-f] #steps]`
Runs the reasoning episode after a query has been posed. It is an error to invoke this command when a query has not been posed. Without any options or arguments, `r` runs the reasoning episode to completion—till the query is answered or the reasoning episode has proceeded long enough to conclude that there will be no answer. When `#steps` is specified with the `-f` option, the reasoning episode is forced to run for `#steps` propagation steps (irrespective of whether the query has been answered or

not). If the `-f` option is not specified, the reasoning episode terminates either after `#steps` cycles or after the query has been answered, whichever happens first.

†Since SHRUTI-CM5 runs reasoning episodes asynchronously, this command does not support the `-f` and/or `#steps` arguments.

Reset Network Syntax: `z [ -q | -v ]`
Resets the network and removes all activation including the query. With the `-v` option, a message is printed out indicating that the network has been reset (default). The message can be suppressed by using the `-q` option.

Set Phases Syntax: `p [#phases]`
Sets the number of phases per clock cycle to `#phases`. The current number of phases is printed out if the command is invoked without an argument.

Display Syntax: `d { -p | -c } name`
Displays the current instantiations of the predicate (with the `-p` option) or concept (with the `-c` option) specified by `name`. An error message is printed if the named predicate or concept is not present in the system.

†Syntax: `d { -p name | -c name }*`
SHRUTI-CM5 supports multiple `-p` and/or `-c` options.

Statistics Syntax: `s [ -a | -k | -q | -c | -s ]`
Prints out knowledge base and reasoning episode statistics. When the system is configured for detailed statistics collection, this command will print out more information. The `-a` option prints out all the accumulated data (default). The `-k` option prints out information about the knowledge base. All details about the current reasoning episode are printed out by the `-q` option. The `-c` and `-s` options print out cumulative data and data from the last propagation step respectively, for the current query.

†Due to the asynchronous nature of the SHRUTI-CM5 system, a global propagation step is not well defined. Hence, SHRUTI-CM5 does not support the `-c` and `-s` options.

†**Display Tagged Activation** Syntax: `a -f first-tag [-l last-tag]`
Displays the number of active predicates and entities with tag values in the specified range. If the `-l` option is not specified, active predicates and entities with tag value equal to `first-tag` are printed.

†**Display Processor Load** Syntax: `l [ -a | -k | -q | -t ] [-n processor]`
Prints out the processor load for the current reasoning episode. When the system is configured for detailed statistics collection, this command will print out more information. The `-a` option prints out all information (default). The `-k` option prints out the distribution of the knowledge base on the processing elements. The distribution of active elements for the current reasoning episode are printed out by the `-q` option. The timing for individual processors (for the current reasoning episode) is displayed by the `-t` option. If the `-n` option is given, required information is displayed for the specified processor. If the `-n` option is not used, data is displayed for all processors in the partition.

References

Ajjanagadde, V. and Shastri, L. (1991). Rules and variables in neural nets. *Neural Computation*, 3:121–134.

Barnden, J. A. and Pollack, J. B., editor (1991). *Advances in Connectionist and Neural Computation Theory, Volume 1*. Ablex Publishing Corporation, Norwood, NJ.

Barnden, J. A. and Srinivas, K. (1991). Encoding techniques for complex information structures in connectionist systems. *Connection Science*, 3(3):269–315.

Charniak, E. (1983). Passing markers: A theory of contextual inference in language comprehension. *Cognitive Science*, 7(3):171–190.

Eicken, T. v., Culler, D. E., Goldstein, S. C., and Schauser, K. E. (1992). Active messages: A mechanism for integrated communication and computation. In *Proceedings of the Nineteenth International Symposium on Computer Architecture*. ACM Press.

Evett, M. P., Andersen, W. A., and Hendler, J. A. (1993). Massively parallel support for efficient knowledge representation. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, pages 1325–1330.

Fahlman, S. E. (1979). *NETL: A System for Representing and Using Real World Knowledge*. MIT Press, Cambridge MA.

Feldman, J. A. and Ballard, D. H. (1982). Connectionist models and their properties. *Cognitive Science*, 6(3):205–254.

Hillis, W. D. (1985). *The Connection Machine*. MIT Press, Cambridge, MA.

Jurafsky, D., Wooters, C., Tajchman, G., Segal, J., Stolcke, A., Fosler, E., and Morgan, N. (1994a). The Berkeley restaurant project. In *Proceedings of the International Conference on Speech and Language Processing*, Yokohama, Japan. To appear.

Jurafsky, D., Wooters, C., Tajchman, G., Segal, J., Stolcke, A., and Morgan, N. (1994b). Integrating advanced models of syntax, phonology, and accent/dialect with a speech recognizer. In *AAAI Workshop on Integrating Speech and Natural Language Processing*, Seattle. To appear.

Lakoff, G. and Johnson, M. (1980). *Metaphors We Live By*. University of Chicago Press, Chicago.

Lange, T. E. and Dyer, M. G. (1989). High-level inferencing in a connectionist network. *Connection Science*, 1(2):181–217.

Leighton, F. T. (1992). *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes*. Morgan Kaufmann, San Mateo, CA.

Lenat, D. B., Guha, R. V., et al. (1990). CYC: Towards programs with common sense. *Communications of the ACM*, 33(8):30–49.

Mani, D. R. (1994). A mathematical analysis of message passing (MIMD) SHRUTI simulation systems. Technical report, University of Pennsylvania. Forthcoming.

Mani, D. R. and Shastri, L. (1993). Reflexive reasoning with multiple instantiation in a connectionist reasoning system with a type hierarchy. *Connection Science*, 5(3 & 4):205–242.

Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K., and Tengi, R. (1990). Five papers on WordNet. Technical Report CSL-43, Princeton University. Revised March 1993.

Moldovan, D. I. (1989). RUBIC: A multiprocessor for rule-based systems. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(4):699–706.

Moldovan, D. I. (1993). *Parallel Processing: From Applications to Systems*. Morgan Kaufmann, San Mateo, CA.

Moldovan, D. I., Lee, W., Lin, C., and Chung, M. (1992). SNAP: Parallel processing applied to AI. *Computer*, 25(5):39–50.

Newell, A. (1992). Unified theories of cognition and the role of Soar. In Michon, J. A. and Akyürek, A., editor, *Soar: A Cognitive Architecture in Perspective*, pages 25–79. Kluwer Academic, Netherlands.

Porter, B., Lester, J., Murray, K., Pittman, K., Souther, A., Acker, L., and Jones, T. (1988). AI research in the context of a multifunctional knowledge base: The botany knowledge base project. Technical Report AI88-88, University of Texas.

Shastri, L. (1991). Why semantic networks? In Sowa, J. F., editors, *Principles of Semantic Networks: Explorations in the Representation of Knowledge*. Morgan Kaufmann, San Mateo, CA.

Shastri, L. and Ajjanagadde, V. (1993). From simple associations to systematic reasoning: A connectionist representation of rules, variables and dynamic bindings using temporal synchrony. *Behavioral and Brain Sciences*, 16(3):417–494.

Sun, R. (1991). Connectionist models of rule-based reasoning. In *Proceedings of the Thirteenth Cognitive Science Conference*, Chicago, IL.

TMC (1991a). Connection machine CM-200 technical summary. Technical Report CMD-TS200, Thinking Machines Corporation, Cambridge, MA.

TMC (1991b). Connection machine CM-5 technical summary. Technical Report CMD-TS5, Thinking Machines Corporation, Cambridge, MA.

TMC (1993). *CMMD Reference Manual. Version 3.0*. Thinking Machines Corporation, Cambridge, MA.

TMC (1994). *CM-5 User's Guide. CMost Version 7.3*. Thinking Machines Corporation, Cambridge, MA.

Waltz, D. L. and Pollack, J. B. (1985). Massively parallel parsing: A strongly interactive model of natural language interpretation. *Cognitive Science*, 9(1):51–74.

Published in the Proceedings of the International Joint Conference on Artificial Intelligence, 1993

A Computational Model of Tractable Reasoning — taking inspiration from cognition*

Lokendra Shastri
Department of Computer & Information Science
University of Pennsylvania
Philadelphia, PA 19104, USA
(215) 898-2661; shastri@cis.upenn.edu

Abstract

Polynomial time complexity is the usual 'threshold' for distinguishing the tractable from the intractable and it may seem reasonable to adopt this notion of tractability in the context of knowledge representation and reasoning. It is argued that doing so may be inappropriate in the context of common sense reasoning underlying language understanding. A more stringent criteria of tractability is proposed. A result about reasoning that is tractable in this stronger sense is outlined. Some unusual properties of tractable reasoning emerge when the formal specification is grounded in a neurally plausible architecture.

1 Introduction

Understanding language is a complex task. It involves among other things, carrying out inferences in order to establish referential and causal coherence, generate expectations, and make predictions. Nevertheless we can understand language at the rate of ***several hundred words per minute*** [Carpenter and Just, 1977]. This rapid rate of language understanding suggests that we can (and do) perform a wide range of inferences very rapidly, automatically and without conscious effort — as though they are a *reflex* response of our cognitive apparatus. In view of this such reasoning may be described as *reflexive* [Shastri, 1991].

As an example of reflexive reasoning consider the sentence 'John seems to have suicidal tendencies, he has joined the Columbian drug enforcement agency.' We can understand this sentence spontaneously and without any deliberate effort even though, doing so involves the use of background knowledge and reasoning. Informally, this reasoning may be as follows: joining the Columbian drug enforcement agency has dangerous consequences, and as John may be aware of this, his decision to join the agency suggests that he has suicidal tendencies. As another example of reflexive reasoning consider the inference 'John owns a car' upon hearing 'John bought a Rolls-Royce'. We can make this inference effortlessly even though it requires multiple steps of inference using background knowledge such as Rolls-Royce is a car and if x buys y then x owns y.

Not all reasoning is, and as complexity theory tells us, cannot be, reflexive. We contrast reflexive reasoning with *reflective* reasoning — reasoning that requires reflection, conscious deliberation, and at times, the use of external props such as paper and pencil (e.g., solving logic puzzles, doing cryptarithmetic, or planning a vacation).

2 Reflexive reasoning necessitates a strong notion of tractability

In order to quantify the notion of reflexive reasoning introduced above, let us make a few observations about such reasoning.

- *Reflexive reasoning occurs with respect to a large body of background knowledge.* A serious attempt at compiling common sense knowledge suggests that our background knowledge base may contain as many as 10^7 to 10^8 items [Guha and Lenat, 1990]. This should not be very surprising given that this knowledge includes, besides other things, our knowledge of naive physics and naive psychology; facts about ourselves, our family, friends, colleagues, history and geography; our knowledge of artifacts, sports, art, music; some basic principles of science and mathematics; and our models of social, civic, and political interactions.
- Items in the background knowledge base are fairly stable and persist for a long-time once they are acquired. Hence this knowledge is best described as *long-term* knowledge and we will refer to this body of knowledge as the long-term knowledge base (LTKB).
- Episodes of reflexive reasoning are triggered by 'small' inputs. In the context of language understanding, an input (typically) corresponds to a sentence that would map into a small number of assertions. For example, the input 'John bought a Rolls Royce' maps into just one assertion (or a few, depending on the underlying representation). The critical observation is that *the size of the input,* $|In|$,

*This work was supported by NSF grants IRI 88-05465 and the ARO grant DAAL 03-89-C-0031.

is insignificant compared to the size of the long-term knowledge base $|LTKB|$.[1] [2]

- The vast difference in the magnitude of $|LTKB|$ (about 10^8) and $|In|$ (a few) becomes crucial when analyzing the tractability of common sense reasoning. Given the actual values of $|In|$ that occur during common sense reasoning, there is a distinct possibility that the overall cost of a derivation may be dominated by the "fixed" contribution of $|LTKB|$. Thus we cannot ignore the cost attributable to $|LTKB|$ and ***we must analyze the complexity of reasoning in terms of*** $|LTKB|$ ***as well as*** $|In|$.

In view of the magnitude of $|LTKB|$, even a cursory analysis suggests that any inference procedure whose time complexity is quadratic or worse in $|LTKB|$ cannot provide a plausible computational account of reflexive reasoning. However, a process that is polynomial in $|In|$ remains viable.

2.1 Time complexity of reflexive reasoning

Observe that although the size of a person's $|LTKB|$ increases considerably from, say, age seven to thirty, the time taken by a person to understand natural language does not. This suggests that the time taken by an episode of reflexive reasoning does not depend on the $|LTKB|$. In view of this it is proposed that a realistic criteria of tractability for reflexive reasoning is one where the time taken by an episode of reflexive reasoning is *independent of* $|LTKB|$ and only depends on the depth of the derivation tree associated with the inference.[3]

2.2 Space complexity of reflexive reasoning

The expected size of the LTKB also rules out any computational scheme whose space requirement is quadratic (or higher) in the size of the KB. For example, the brain has only about 10^{12} cells most of which are involved in processing of sensorimotor information. Hence even a linear space requirement is fairly generous and leaves room only for a modest 'constant of proportionality'. In view of this, it is proposed that the admissible space requirement of a model of reflexive reasoning be *no more than linear in* $|LTKB|$.

To summarize, it is proposed that as far as (reflexive) reasoning underlying language understanding is concerned, the appropriate notion of tractability is one where

- the reasoning time is independent of $|LTKB|$ and is only dependent on $|In|$ and the depth of the derivation tree associated with the inference, and
- the associated space requirement, i.e., the space required to encode the LTKB plus the space required to hold the working memory during reasoning should be no worse than *linear* in $|LTKB|$.

In spite of the apparent significance of reflexive reasoning there have been very few attempts at developing a computational account of such inference. Some past exceptions being Fahlman's work on NETL [1979] and Shastri's work on a connectionist semantic memory [1988]. However these models dealt primarily with inheritance and classification within an *IS-A* hierarchy. Hölldobler [1990] and Ullman and van Gelder [1988] have proposed parallel systems for performing quite complex logical inferences, however, both these systems have unrealistic space requirements. The number of nodes in Hölldobler's system is quadratic in the the size of the knowledge base (KB) the number of processors required by Ullman and van Gelder is even higher. Ullman and van Gelder treat the number of nodes required to encode the background KB as a fixed cost, and hence, do not refer to its size in computing the space complexity of their system. If the size of such a KB is taken into account, the number of processors required by their system turns out to be a high degree polynomial.

A significant amount of work has been done by researchers in knowledge representation and reasoning to identify classes of limited inference that can be performed efficiently (e.g., see [Frisch and Allen, 1982]; [Brachman and Levesque, 1984]; [Patel-Schneider, 1985]; [Dowling and Gallier, 1984]; [Levesque, 1988]; [Selman and Levesque, 1989]; [McAllester, 1990]; [Bylander *et al.*, 1991]; [Kautz and Selman, 1991]). This work has covered a wide band of the complexity spectrum but none that meets the strong tractability requirement discussed above. Most results stipulate polynomial time complexity, restrict inference in implausible ways (e.g., by excluding chaining of rules), and/or deal with limited expressiveness (e.g., deal only with propositions).

3 A tractable reasoning class

Below we describe a class of reasoning that is tractable with reference to the criteria stated above. The characterization of such a class is different (but analogous) for forward and backward reasoning. In this paper we will focus on backward reasoning.

Some definitions:

Let us define *rules* to be first-order sentences of the form:

$\forall x_1, ..., x_m \, [P_1(...) \wedge P_2(...) ... \wedge P_n(...) \Rightarrow \exists z_1, ... z_l \, Q(...)]$

where the arguments of P_i's are elements of $\{x_1, ... x_m\}$, and an argument of Q is either an element of $\{x_1, ... x_m\}$, an element of $\{z_1, ... z_l\}$, or a constant. □

Any variable that occurs in multiple argument positions in the antecedent of a rule is a *pivotal* variable. □

[1] A small input may, however, lead to a potentially large number of elaborate inferences. For example, the input 'John bought a Rolls-Royce' may generate a number of reflexive inferences such as 'John bought a car', 'John owns a car', 'John has a driver's license', 'John is perhaps a wealthy man', etc.

[2] Some of these inferences may be 'soft' inferences, but the issue of deductive versus evidential nature of inferences is irrelevant to our current concerns.

[3] The restriction that the reasoning time be independent of $|LTKB|$ may seem overly strong and one might argue that perhaps logarithmic time may be acceptable. Our belief that the stronger notion of effectiveness is relevant, however, is borne out by results which demonstrate that there does exists a class of reasoning that can be performed in time independent of $|LTKB|$.

Note that the notion of a pivotal variable is local to a rule.

A rule is *balanced* if all pivotal variables occurring in the rule also appear in its consequent. □

For example, the rule $\forall x, y, z\ P(x,y) \wedge R(x,z) \Rightarrow S(y,z)$ is not balanced because the pivotal variable x does not occur in the consequent. On the other hand, the rule $\forall x, y, z\ P(x,y) \wedge R(x,z) \Rightarrow S(x,z)$ is balanced because the pivotal variable x does occur in the consequent. The fact that y does not appear in the consequent is immaterial because y occurs only once in the antecedent and hence, is not a pivotal variable.

Facts are partial or complete instantiations of predicates. Thus facts are atomic formulae of the form $P(t_1, t_2 ... t_k)$ where t_i's are either constants or distinct existentially quantified variables. □

Queries have the same form as facts. Let us distinguish between *yes-no* queries and *wh*-queries. A query, all of whose arguments are bound to constants corresponds to the *yes-no* query: 'Does the query follow from the rules and facts encoded in the long-term memory of the system?' A query with existentially quantified variables, however, has several interpretations. For example, the query $P(a,x)$, where a is a constant and x is an existentially quantified argument, may be viewed as the *yes-no* query: 'Does $P(a,x)$ follow from the rules and facts for some value of x?' Alternately this query may be viewed as the *wh*-query: 'For what values of x does $P(a,x)$ follow from the rules and facts in the system's long-term memory?' □

Consider a query Q and a LTKB consisting of facts and balanced rules. A derivation of Q obtained by backward chaining is *threaded* if all pivotal variables occurring in the derivation get bound and their bindings can be traced back to the bindings introduced in Q. □

Given a LTKB consisting of facts and balanced rules, a *reflexive* query is one for which there exists a threaded proof. □

3.1 A class of tractable reasoning

The worst-case time for answering a reflexive *yes-no* query, Q, is proportional to $V|In|^V d$, where:

- $|In|$ is the number of *distinct* constants in Q.
- V is as follows: Let V_i be the arity of the predicate P_i. Then V equals $max(V_i)$, i ranging over all the predicates in the LTKB.
- d equals the depth of the shallowest derivation of Q given the LTKB.

Observe that the worst-case time is i) *independent* of $|LTKB|$, ii) polynomial in $|In|$ and iii) only proportional to d.

As observed in Section 2, while $|LTKB|$ may be as much as 10^8, $|In|$ is simply the number of (distinct) 'entities' referred to in the input. In the context of natural language understanding, $|In|$ would be quite small (typically, less than 5). We also expect V, the maximum arity of predicates in the LTKB to be quite small.

An answer to a *wh*-query can also be computed in time proportional to $V|In|^V d$, except that $|In|$ now equals the arity of the query predicate Q.

The *space* requirement is *linear* in $|LTKB|$ and polynomial in $|In|$. This includes the cost of encoding the LTKB as well as the cost of maintaining the dynamic state of the 'working memory' during reasoning.

An informal explanation of the result

The number of times a predicate P may get instantiated in a threaded derivation of a query cannot exceed $|In|^V$. This follows from the observation that P has at most V arguments and each of these can get bound to at most $|In|$ distinct constants. Since each predicate instantiation can contain at most V bindings, the propagation of argument bindings from one predicate to another can be carried out in time proportional to $V|In|^V$. This assumes that the correspondence (specified by the rules in the LTKB) between the arguments of the antecedent and consequent predicates are hard-wired.

It can be shown that the propagation of argument bindings from multiple predicates to a predicate can be carried out in parallel (see [Mani and Shastri, 1992] for a possible implementation of such a parallel binding propagation scheme). This means that the time required to carry out one step of a parallel breadth-first derivation is only proportional to $V|In|^V$. It follows that the time required to carry out a d step parallel derivation is proportional to $V|In|^V d$.

Lower bound nature of above result

In general, derivations that involve unbalanced rules or those that do not satisfy the *threaded* property cannot be computed in time independent of $|LTKB|$, if the available space is no more than *linear* in $|LTKB|$ [Dietz *etal.*, 1993]. This result follows from the observations that i) the *common-element* problem, i.e., the problem of determining whether two sets share a common element, can be reduced to the problem of computing a derivation involving unbalanced rules and/or non-threaded derivations, ii) the *sorting* problem can be reduced to the common-element problem, and iii) the lower bound on the sorting problem is *nlogn* (where n would corresponds to $|LTKB|$). Thus derivations involving unbalanced rules and non-threaded derivations may not be computed in time independent of $|LTKB|$ unless one makes use of more than *linear* space.

3.2 Worst-case versus expected case

The above result offers a worst-case characterization which assumes that during the derivation, *all variables will get instantiated with all possible bindings involving constants in Q*. This will not be the case in a typical situation. In fact it may be conjectured that in a typical episode of reasoning, the actual time will seldom exceed $50d$ (see next section).

4 A neurally motivated model of tractable reasoning

We have proposed a neurally plausible model (SHRUTI) that can encode a LTKB of the type described above, together with a term hierarchy and perform a class of forward as well as backward reasoning with extreme efficiency [Shastri and Ajjanagadde, 1990]; [Ajjanagadde and Shastri, 1991]; [Mani and Shastri, 1991]; [Mani and Shastri, 1992]; [Shastri, 1992]. SHRUTI can draw inferences in time that is only proportional to the *depth* of the shallowest derivation leading to the conclusion. A SHRUTI like model has also been used by Henderson [1992] to design a parser for English. The parser's speed is independent of the size of the lexicon and the grammar, and it offers a natural explanation for a variety of data on long distance dependencies and center embedding.

If we set aside SHRUTI's ability to perform terminological reasoning, the class of reasoning that SHRUTI can perform efficiently is a subclass of the class of reasoning specified in the previous section. The additional restrictions placed on SHRUTI's reasoning ability are motivated by gross constraints on the speed at which humans can perform reflexive reasoning and gross neurophysiological parameters such as:

1. π_{max}, the maximum period at which nodes can be expected to sustain synchronous activity,
2. ω, the tolerance or the minimum lead/lag that must be allowed between the spiking of two nodes that are firing in synchrony,
3. the time it takes a cluster of synchronous nodes to drive a connected cluster of nodes to fire in synchrony.

The details of the model are beyond the scope of this paper and the reader is referred to [Shastri and Ajjanagadde, 1990]. Let us however, state the additional constraints on the class of reasoning SHRUTI can perform.

4.1 Additional constraints on the reasoning performed by SHRUTI

SHRUTI can encode a LTKB of facts and *balanced* rules and answer yes to any *reflexive yes-no* query in time proportional to the *depth* of the shallowest derivation leading to a derivation of the query provided:

1. The number of distinct constants specified in the query does not exceed k_1, where k_1 is bounded by π_{max}/ω) (biological data suggests that k_1 is small, perhaps between 5 and 10).

   The model suggests that as long as the number of entities introduced by the query is 5 or less, there will essentially be no cross-talk among the facts inferred during reasoning. If more than 5 entities occur, the window of synchrony would have to shrink appropriately in order to accommodate all the entities. As this window shrinks, the possibility of cross-talk between bindings would increase until eventually, the cross-talk would become excessive and disrupt the system's ability to perform systematic reasoning. The biological data suggests that a neurally plausible *upper bound* on the number of distinct entities that can occur in the reasoning process is about 10. Of course, these entities may occur in multiple facts and participate in a number of inferences.

   It may be significant that the bound on the number of entities that may be referenced by the active facts during a derivation relates well to 7 ± 2, the robust measure of short-term memory capacity [Miller, 1956]. Note however, that SHRUTI does not place a small limit on the number of *facts* that can be simultaneously active — indeed a very large number of facts can be involved in each derivation carried out by SHRUTI.

2. During the processing of the query, each predicate may only be instantiated at most k_2 times.

   Note that this restriction only applies to run-time or 'dynamic' instantiations of predicates and not to 'long-term' facts stored in the system. As argued in [Shastri, 1992] a plausible values of k_2 is somewhere between 3–5. Also, k_2 need not be the same for all predicates. The application of a SHRUTI-like model to parsing by Henderson also suggests that a value of k_2 under 3 may be sufficient for parsing English sentences.

Some typical retrieval and inference timings

If we set system parameters of SHRUTI to some neurally motivated values, SHRUTI demonstrates that a system made up of simple and slow neuron-like elements can perform a wide range of inferences (both forward, backward and those involving a type hierarchy) within a few hundred milliseconds.

If we choose the period of oscillation of nodes to be 20 milliseconds, assume that nodes can synchronize within two periods of oscillations and pick k_2 equal to 3, SHRUTI takes 320 milliseconds to infer 'John is jealous of Tom' after being given the dynamic facts 'John loves Susan' and 'Susan loves Tom' (this involves the rule 'if x loves y and y loves z then x is jealous of z). The system takes 260 milliseconds to infer 'John is a sibling of Jack' given 'Jack is a sibling of John' (this involves the rule 'if x is a sibling of y then y is a sibling of x). Similarly, the system takes 320 milliseconds to infer 'Susan *owns* a car' after its internal state is initialized to represent 'Susan *bought* a Rolls-Royce' (using the rule 'if x buys y then x owns y' and the *IS-A* relation, 'Rolls-Royce is a car').

If SHRUTI's long-term memory contains the fact 'John bought a Rolls-Royce', SHRUTI takes 140 milliseconds, 420 milliseconds, and 740 milliseconds, respectively, to answer 'yes' to the queries 'Did John buy a Rolls-Royce', 'Does John own a car?' and 'Can John sell a car?' (the last query also makes use of the rule 'if x owns y then x can sell y). Note that the second and third queries also involve inferences using rules as well as *IS-A* relations.

The above times are independent of $|LTKB|$ and do not increase when additional rules, facts, and *IS-A* relationships are added. If anything, these times may decrease if a new rule is added that leads to a shorter inference path.

5 Conclusion

We have proposed a criteria for tractable reasoning that is appropriate in the context of common sense reasoning underlying language understanding. We have suggested that an appropriate measure of tractability for such reasoning is one where the time complexity is independent of, and the space complexity is no more than linear in, the size of the long-term knowledge base. We have also identified a class of reasoning that is tractable in this sense. This characterization of tractability can be further refined by cognitive and biological considerations. This work suggests that the expressiveness and the inferential ability of a representation and reasoning systems may be limited in unusual ways to arrive at extremely efficient yet fairly powerful knowledge representation and reasoning systems.

References

[Ajjanagadde and Shastri, 1991] V.G. Ajjanagadde and L. Shastri. Rules and variables in neural nets. *Neural Computation*, 3:121-134.

[Brachman and Levesque, 1984] R. Brachman and H. Levesque. The tractability of Subsumption in frame-based description languages. In *Proceedings of AAAI-84*, the fourth national conference on artificial intelligence. Morgan Kaufman.

[Bylander *et al.*, 1991] T. Bylander, D. Allemang, M. C. Tanner, J. R. Josephson. The computational complexity of abduction. *Artificial Intelligence*, 47(1-3), 25–60.

[Carpenter and Just, 1977] P.A. Carpenter and M.A. Just. Reading Comprehension as Eyes See It. In: *Cognitive Processes in Comprehension.*

[Dietz *etal.*, 1993] P. Dietz, D. Krizanc, S. Rajasekaran, L. Shastri. A lower-bound result for the common element problem and its implication for reflexive reasoning. Forthcoming Technical Report, Department of Computer and Information Science, Univ. of Pennsylvania.

[Dowling and Gallier, 1984] W.F. Dowling and J.H. Gallier. Linear time algorithm for testing the satisfiability of propositional horn formula. *Journal of Logic Programming*, 3:267-284.

[Fahlman, 1979] S.E. Fahlman. *NETL: A system for representing real-world knowledge*, MIT Press.

[Frisch and Allen, 1982] A.M. Frisch and J.F. Allen. Knowledge retrieval as limited inference. In: *Notes in Computer Science: 6th Conference on Automated Deduction* ed. D. W. Loveland. Springer-Verlag.

[Guha and Lenat, 1990] R.V. Guha and D.B. Lenat. Cyc: A Mid-Term report. *AI Magazine*, Volume 11, Number 3, 1990.

[Henderson, 1992] J. Henderson. A Connectionist Parser for Structure Unification Grammar. In *Proceedings of ACL-92.*

[Hölldobler, 1990] S. Hölldobler. CHCL: A Connectionist Inference System for Horn Logic based on the Connection Method and Using Limited Resources. *TR-90-042*, International Computer Science Institute, Berkeley, CA.

[Kautz and Selman, 1991] H.A. Kautz and B. Selman. Hard problems for Simple Default Logics. *Artificial Intelligence*, 47(1-3), 243-279.

[Levesque, 1988] H.J. Levesque. Logic and the complexity of reasoning. *Journal of Philosophical Logic*, 17, pp 335-389.

[Mani and Shastri, 1992] D.R. Mani and L. Shastri. A connectionist solution to the multiple instantiation problem using temporal synchrony. In *Proceedings of the Fourteenth Conference of the Cognitive Science Society.* Lawrence Erlbaum.

[Mani and Shastri, 1991] D.R. Mani and L. Shastri. Combining a Connectionist Type Hierarchy with a Connectionist Rule-Based Reasoner. In *Proceedings of the Thirteenth Conference of the Cognitive Science Society.* Lawrence Erlbaum.

[McAllester, 1990] D.A. McAllester. Automatic recognition of tractability in inference relations. Memo 1215, MIT Artificial Intelligence Laboratory, February 1990.

[Miller, 1956] G.A. Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *The Psychological Review*, 63(2), pp. 81-97.

[Patel-Schneider, 1985] P. Patel-Schneider. A decidable first-order logic for knowledge representation. In *Proceedings of IJCAI-85.* Morgan Kaufman.

[Rajasekaran, 1992] S. Rajasekaran. Personal communication.

[Selman and Levesque, 1989] B. Selman and H.J. Levesque. The tractability of path-based inheritance. In *Proceedings of IJCAI-89.* pp. 1140-1145. Morgan Kaufmann.

[Shastri, 1992] L. Shastri. Neurally motivated constraints on a working memory capacity of a production system for rapid parallel processing. To appear in the *Proceedings of the Fourteenth Conference of the Cognitive Science Society.* Lawrence Erlbaum.

[Shastri, 1991] L. Shastri. Why Semantic networks? In *Principles of Semantic Networks.* Edited by John Sowa. Morgan Kaufman Los Altos.

[Shastri, 1988] L. Shastri. *Semantic networks : An evidential formulation and its connectionist realization*, Pitman London/ Morgan Kaufman Los Altos. 1988.

[Shastri and Ajjanagadde, 1990] L. Shastri and V.G. Ajjanagadde. From Simple Associations to Systematic Reasoning: A connectionist representation of rules, variables and dynamic bindings using temporal synchrony. Technical Report MS-CIS-90-05, Department of Computer and Information Science, Univ. of Pennsylvania. (Revised January 1992). To appear in *Behavioral and Brain Sciences.*

[Ullman and van Gelder, 1988] J.D. Ullman and A. van Gelder. Parallel Complexity of Logical Query Programs. *Algorithmica*, 3, 5-42.

To appear in Handbook of Brain Theory and Neural Networks, Ed. Michael Arbib MIT Press

Structured Connectionist Models

Lokendra Shastri
International Computer Science Institute
1947 Center St. Suite 600
Berkeley, CA 94704

RUNNING HEAD: Structured Connectionist Models

Correspondence:

Lokendra Shastri

International Computer Science Institute

1947 Center St. Suite 600

Berkeley, CA 94704

Phone: (510) 643-9153

Fax: (510) 643-7684

email: shastri@icsi.berkeley.edu

1. INTRODUCTION

Artificial intelligence (AI) and cognitive science have made considerable advances over the past four decades, but it is widely believed that solutions resulting from the "classical" approach to these disciplines lack scalability, gradedness, robustness, and adaptability. Consider scalability. Although existing AI systems may perform credibly within restricted domains they do not scale up: as the domain grows larger a system's performance degrades drastically and it can no longer solve interesting problems in acceptable time-scales. Consider gradedness. Research in AI and cognitive science has made it apparent that the solution of a cognitive task emerges as a result of rich context-sensitive interactions among a large number of graded factors. Classical models, rooted in the von Neumann architecture, are not suited for articulating this view of computation (for a discussion of robustness and adaptability see SYMBOLS IN NEURAL REPRESENTATIONS).

Research in connectionism is motivated by the belief that in order to address the limitations mentioned above, one must pay attention to the computational characteristics of the brain. After all, the brain is the only existing physical system that exhibits the requisite attributes, and it seems reasonable to expect that identifying neurally motivated constraints — albeit, at an abstract computational level — and incorporating them into our models would lead to novel and critical insights.

In addition to recognizing the importance of neurally motivated constraints, the *structured connectionist* approach (Feldman et al., 1988) also recognizes that a number of insights acquired by disciplines such as computer science, AI, psychology, linguistics and learning theory will have to be leveraged in developing solutions to difficult problems in AI and cognitive science. These insights pertain to recognizing the power of problem decomposition, hierarchical processing, and structured representations; the need for representational and inferential adequacy; and the role of complexity analysis.

A key difference between the structured connectionist approaches and the so called *dis-*

tributed approach is as follows: The fully distributed approach assumes that each "item" (a concept or mental object) is represented as a pattern of activity distributed over a *common* pool of nodes (van Gelder, 1992). This notion of representation suffers from several fundamental limitations. Consider the representation of 'John and Mary'. If 'John' and 'Mary' are represented as patterns of activity over the entire network such that each node in the network has a specific value in the patterns for 'John' and 'Mary', respectively, then how can the network represent 'John' and 'Mary' at the same time? The situation gets even more complex if the system has to represent relations such as 'John loves Mary', or 'John loves Mary but Tom loves Susan'. In contrast to the distributed approach, the structured approach holds that small clusters of nodes can have distinct representational status (for simplicity, structured connectionist models often equate a small cluster of nodes with a single *idealized* node). In particular, there exist small clusters of nodes that act as 'focal' nodes or 'handles' of *learned* concepts and provide access to more elaborate node structures which make up the detailed encoding of concepts. Such a detailed encoding might include various features of the concept as well as its relationship to other concepts (see (Feldman, 1989) and the article by Shastri in (Barnden and Pollack, 1991)). The fully distributed view is also inconsistent with the continually emerging data about the localization of function in the brain.

The structured approach is often wrongly equated with the so called "grandmother cell" approach which assumes that each concept is represented by a distinct node. This misunderstanding stems from an incorrect interpretation of the representational role of 'focal' nodes.

A second difference between the structured and the fully distributed approaches concerns learning. The latter underplays the importance of structure and assumes that essentialy all the required structure emerges as a result of general-purpose learning processes operating on relatively unstructured hidden layers. The structured approach holds that such a *tabula-rasa* view is untenable on grounds of computational complexity; training unstructured networks using general purpose learning techniques is not a feasible methodology for obtaining scalable solutions to complex problems. The structured approach emphasizes the importance of prior structure for effective learning and requires that the initial design of network models — for example, the broad representational significance of nodes, the number of representational levels in the network, and the network interconnection pattern — reflect the structure of the problem.

2. SOME NEURAL CONSTRAINTS ON COGNITIVE MODELS

2.1. Representational constraints

With over 10^{11} computing elements and 10^{15} interconnections, the human brain's capacity for encoding, communicating, and processing information seems awesome. But if the brain is extremely powerful it is also extremely limited: First, neurons are slow computing devices. Second, although the spatio-temporal integration of inputs performed by neurons is quite complex, it is relatively undifferentiated with reference to the needs of symbolic computation. Third, neurons communicate via relatively simple 'messages' that can encode only a few bits of information. Hence a neuron's output cannot be expected to encode names, pointers, or complex structures.

A specific limitation of neurally plausible systems is that they have difficulty representing composite structures in a *dynamic* fashion (also see COMPOSITIONALITY IN NEURAL SYSTEMS). Consider the representation of the fact *give(John, Mary, Book1).* This fact cannot be represented dynamically by simply activating the roles *giver*, *recipient*, and *give-object*, and the constituents 'John', 'Mary', and 'Book1'. Such a representation would be indistinguishable from the representation of *give(Mary, John, Book1).* The problem is that representing a fact requires representing the appropriate *bindings* between roles and their fillers. It is easy to represent static (long-term) bindings using dedicated nodes and links. For example, one could posit a separate 'binder' node for each role-filler pair to represent role-filler bindings. Such a scheme is adequate for representing long-term knowledge because the required binder nodes may be recruited over time. This scheme however, is implausible for representing dynamic bindings arising during language understanding and visual processing since it is unlikely that there exist mechanisms for establishing new links within such time scales. The alternative that interconnections between *all possible* pairs of roles and fillers already exist and the appropriate ones become "active" temporarily to represent dynamic bindings is also ruled out given the prohibitively large number of such role-filler bindings. Techniques for representing bindings based on the von Neumann architecture cannot be used since the storage and processing capacity of nodes and the resolution of their outputs is insufficient to store and communicate names or pointers.

2.2. Scalability in time

We can visually recognize items from a potential pool of 100,000 commonplace items in about a hundred milliseconds and can understand language at the rate of several words per second, even though doing so involves perceptual processing, lexical access, parsing, and reasoning. This indicates that we can perform a wide range of visual, linguistic, and inferential tasks within a few hundred milliseconds. This observation provides a powerful constraint that can inform our search for cognitive models (Feldman and Ballard, 1982).

2.3. Scalability in space

Although the number of neurons in the brain is quite large, it is not too large compared to the 'size' of the problems it must solve! Consider vision and reasoning. The retinal output consists of a million signals and similarly, our common sense knowledge base may contain more than a million items. This suggests that any model of vision or reasoning whose node requirement grows quadratically or higher with respect to the size of the problem, may not be neurally plausible.

2.4. From constraints to predictions

While cognitive agents solve a wide range of tasks with extreme efficiency their cognitive ability is also limited in a number of ways — examples abound in vision, language, and short-term memory. It is expected that structured connectionist models that incorporate representational and scalability constraints discussed above would help in understanding and explaining not only the strengths of human cognition but also some of its limitations.

3. SOME STRUCTURED CONNECTIONIST MODELS

3.1. Early work

One of the earliest examples of a structured connectionist model was the interactive activation model for letter perception by McClelland and Rumelhart (1981). The model consisted of three layers of nodes corresponding to visual letter features. letters, and words. Nodes representing mutually exclusive hypotheses within the letter and word layers inhibited each other. For example, since only one letter may exist in a given letter position, all nodes representing letters in the same position inhibited each other. A node in the feature layer was connected via excitatory connections to nodes in the letter layer representing letters that contained that feature. Similarly, a node in the letter layer was connected via excitatory connections to nodes in the word layer representing words that contained that letter in the appropriate position. Additionally, there were reciprocal connections from the word layer to the letter layer. The interconnection pattern allowed bottom-up perceptual processing to be guided by top-down expectations. The model could explain a number of psychological findings about the preference of words and pronounceable non-words over other non-words and isolated letters.

Other examples of early structured connectionist models were word sense disambiguation models developed by Cottrell and Small (1983) and Waltz and Pollack (1985). Most words have multiple senses but we are able to exploit contextual and syntactic information to rapidly disambiguate the meanings of words. These models demonstrated how such disambiguation might occur. Cottrell and Small's model consisted of a three-level network consisting of the lexical (word) level, the word-sense level, and the case-level. There were inhibitory links between different noun senses of the same word, and between different predicate senses of the same word. A node at the lexical level was connected to all its senses at the word-sense level. Connections between the word-sense level and the case-level expressed all feasible bindings between predicates and objects. As a sentence was input by activating the appropriate lexical items in a sequence, activation flowed through the network and the combination of lexical items, word senses and case assignment that best fit the input formed a stable coalition of active nodes.

Another example of a structured connectionist model was the connectionist semantic network model, CSN (Shastri, 1988). CSN viewed memory as a collection of concepts organized in a *IS-A* hierarchy (e.g., "Bird IS-A Animal') and allowed the attachment of property values to concepts. Unlike a traditional semantic network, a property-value attachment in CSN consisted of distributional information indicating how members of a concept were distributed with respect to the different values of the property. CSN could answer (i) inheritance queries, i.e., infer the *most likely* value of a specified property for a given concept and (ii) recognition queries, i.e., given a description consisting of property-value pairs, find the concept that *best* matched the given description. CSN found answers to queries by combining information encoded in the network in accordance with an evidential formalization based on the principle of maximum entropy. In particular, CSN could use distributional information to deal with with exceptional and conflicting information in a principled manner and disambiguate 'multiple inheritance' situations that could not be dealt with by extant formulations of inheritance in AI.

CSN encoded concepts, properties and values using 'focal' nodes. The IS-A relations were encoded as links, and property values were attached to concepts by connecting the appropriate property, value and concept nodes via binder nodes. The weights on links between concept, value, and binder nodes captured the distributional information associated with a property value attachment. A query was posed by activating appropriate nodes. Thereafter, CSN performed the required inferences automatically by propagating graded activations and combining activations using appropriate activation combination rules.

3.2. Recent models of memory and reasoning

The models above made significant contributions but were limited in their expressive power and inferential ability. One of their key limitations was that they did not address the dynamic binding problem. For example, the McClelland and Rumelhart model required n-fold repetition of letter and feature layers to deal with words of length n; it could not dynamically bind a letter to a position in a word. The Cottrell and Small and Waltz and Pollack systems pre-wired all possible bindings using dedicated nodes and links. Recently there has been significant progress in solving this problem. These include the CONPOSIT system (see article by Barnden and Srinivas in (Barnden and Pollack, 1991)) the ROBIN system (Lange and Dyer, 1989), the SHRUTI system (Shastri and Ajjanagadde, 1993), and the CONSYDERR system (Sun, 1992). We give a brief overview of SHRUTI which shares a number of representational and functional features with ROBIN but differs from it in the mechanism used for representing dynamic bindings.

SHRUTI can encode a large number of specific facts, general rules, as well as IS-A relations between concepts and perform a broad class of reasoning with extreme efficiency. SHRUTI encodes an n-ary predicate as a cluster of nodes which includes n role nodes (refer to Figure 1). Nodes such as *John* and *Mary* correspond to *focal* nodes of the complete representations of the individuals 'John' and 'Mary'. A rule is encoded by linking the roles of the antecedent and consequent predicates in accordance with the correspondence between roles specified in the rule. SHRUTI represents dynamic bindings using *synchronous* firing of the appropriate argument and concept nodes. For example, the dynamic fact *give(John,Mary,Book1)* is

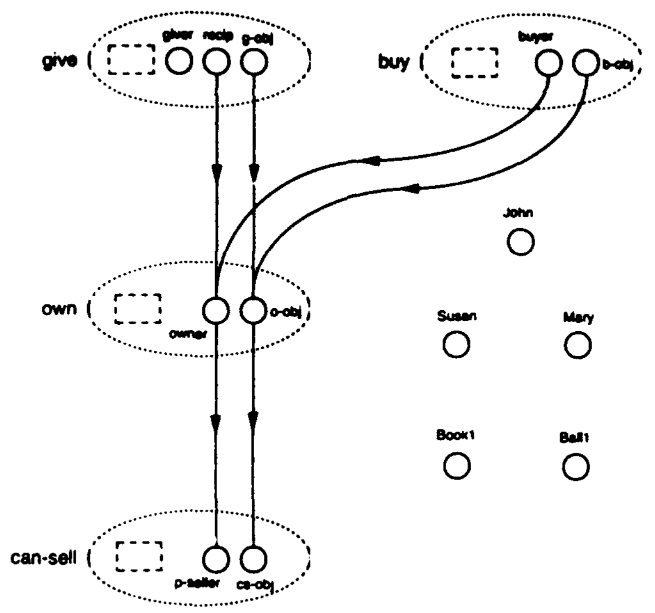


Figure 1: Encoding of predicates, concepts and the rules $\forall x,y,z$ [*give(x,y,z)* $\Rightarrow$ *own(y,z)*], $\forall x,y$ [*buy(x,y)* $\Rightarrow$ *own(x,y)*], and $\forall x,y$ [*own(x,y)* $\Rightarrow$ *can-sell(x,y)*].

represented by a *rhythmic* pattern of activity wherein the focal nodes John, Mary, and Book1 are firing in synchrony with the role nodes *giver*, *recip*, and *g-obj* respectively. By virtue of the interconnections between role nodes of the predicates *give*, *own*, and *can-sell*, this state of activation evolves so that (i) *owner*, and in turn, *p-seller* start firing in synchrony with *recip* and hence, Mary and (ii) *o-obj* and, in turn, *cs-obj* starts firing in synchrony with *g-obj* and hence, Book1. The resulting firing pattern corresponds to the dynamic facts *give(John,Mary,Book1)*, *own(Mary,Book1)*, and *can-sell(Mary,Book1)*. The key assumption here is that if nodes A and B are linked, the firing of A leads to a synchronous firing of B. For more on the role of synchrony and the dynamic binding problem see COMPOSITIONALITY IN NEURAL SYSTEMS.

SHRUTI can encode *long-term* facts and a bounded number of instantiations of each predicate and concept. The latter allows it to deal with reasoning involving 'bounded recursion'. SHRUTI can also represent a type (IS-A) hierarchy and allows categories as well as instances in rules, facts, and queries. By using appropriate weights on links, the system can also encode soft/evidential rules. The time SHRUTI takes to generate a chain of inference is independent of the total number of rules and facts and is equal to $l * \alpha$, where l is the number of steps in the chain of inference and α is the time required for connected nodes to synchronize. If we assume α to be about 100 milliseconds, SHRUTI demonstrates that a system of simple computing elements can encode millions of items and draw interesting inferences in a few hundred *milliseconds. An implementation* of the system on a CM-5 encodes over 300,000 items and responds to queries with derivation lengths of up to 8 in under a second.

Instead of using synchronous firing of nodes to represent and propagate bindings, ROBIN and CONSYDERR assign a distinct "signature" to each concept and propagate these codes to establish bindings. A signature may take the form of a unique activation value or a pattern of activity. CONPOSIT creates bindings by virtue of the relative position of active nodes and the similarity of patterns. The use of temporal synchrony in SHRUTI leads to a number of predictions about the capacity of the *working memory* underlying rapid reasoning (WMRR). Thus SHRUTI predicts that a very large number of facts may be co-active in WMRR and a large number of rules may fire simultaneously as long as: the maximum number of distinct entities that can occur as role-fillers in the dynamic facts is small (at most 10), and only a small number of instances of each predicate ($\approx$ 3) may be co-active at the same time. The temporal approach also predicts that the depth to which an agent may reason rapidly but systematically is bounded. All these constraints are motivated by biological considerations. For example, each entity participating in dynamic bindings occupies a distinct phase and hence, the number of distinct entities that can occur as role-fillers in dynamic facts cannot exceed $\lfloor \pi_{max}/\omega \rfloor$, where π_{max} is the maximum delay between consecutive firings of synchronous cell-clusters, and ω equals the allowable jitter in the firing times of synchronous cell-clusters.

Henderson (1994) has developed an on-line parser for English using a SHRUTI-like architecture. The parser's speed is independent of the size of the grammar and it can recover the structure of arbitrary long sentences as long as the dynamic state required to parse the sentence does not exceed the capacity of the parser's working memory. The parser shows that the constraints on the working memory help explain several properties of human parsing involving long distance dependencies, garden path effects and our limited ability to deal with center-embedding.

3.3. Significance of structure

The representational and inferential power of structured connectionist systems such as SHRUTI and their ability to draw inferences in parallel is directly attributable to their use of structured representations. Any system that uses fully distributed representations will be incapable of representing multiple dynamic facts and applying multiple rules simultaneously. Attempts to develop distributed systems to handle relations invariably end up positing several distinct banks — one for each role — thereby stepping away from a fully distributed mode, or fall back on seriality. It is not surprising that distributed systems such as DCPS (Touretzky and Hinton, 1988) have extremely limited capacity for encoding dynamic structures and are serial at the level of rule-application.

3.4. Learning in structured networks

The models discussed thus far did not address the issue of learning in detail (though an outline of how rule-learning might occur in a SHRUTI-like system appears in (Shastri and Ajjanagadde, 1993)). Regier's (1992) model for learning the lexical semantics of natural language spatial terms provides a concrete example of learning within the structured connectionist paradigm. The model observes movies of simple 2-dimensional objects moving relative to one another — where each movie is labeled as an example of some spatial term from a natural language — and learns the association between the label (word) and the event/relation they describe. The model succesfully learned several spatial terms for diverse natural languages. The model includes structured network components that reflect prior constraints about the task as well as the usual "hidden layers", and demonstrates how structured connectionist networks can incorporate flexible learning ability and at the same time, leverage prior structure to achieve tractability.

In addition to incremental learning driven by repeated exposure to a large body of training data, structured models have also made use of one-shot learning using *recruitment* learning schemes (e.g., see (Shastri, 1988) and article by Diederich in (Barnden and Pollack, 1991)).

4. DISCUSSION

Structured connectionism offers a rich framework for developing models of cognition that are guided by biological, behavioral, and computational constraints. The approach has been productive and resulted in a number of models that are informed by insights from diverse disciplies such as computer science, AI, psychology, linguistics and neuroscience. Having resolved some difficult representational problems, the focus of work is shifting toward the study of structured adaptive networks that are grounded in perception and action.

Acknowledgements

The preparation of this paper was funded by ONR grant N00014-93-1-1149. Thanks to Jerry Feldman, Michael Arbib and an anonymous reviewer for their suggestions.

REFERENCES

* Barnden, J., and Pollack, L., 1991, (Eds) Advances in Connectionist and Neural Computation Theory, vol. 1. Ablex.

Cottrell, G. W., and Small, S. L., 1983, A connectionist scheme for modeling word sense disambiguation. Cognition and Brain Theory, 6:89-120.

* Feldman, J. A., and Ballard, D. H., 1982, Connectionist models and their properties. Cognitive Science, 6(3):205-254.

* Feldman, J. A., 1989, Neural Representation of Conceptual Knowledge, in Neural Connections, Mental Computation, (L. Nadel, L. A. Cooper, P. Culicover, and R. M. Harnish, Eds.), MIT Press.

* Feldman, J. A., Fanty, M. A., and Goddard, N. H., 1988, Computing with Structured Neural Networks. IEEE Computer, March 1988, pp 91-103.

van Gelder, T., 1992, Defining 'Distributed Representation'. Connection Science, 4(3&4):175-191.

Henderson, J., 1994, Connectionist Syntactic Parsing Using Temporal Variable Binding. Journal of Psycholinguistic Research, (to appear).

Lange, T. E., and Dyer, M. G., 1989, High-level Inferencing in a Connectionist Network. Connection Science, 1(2):181-217.

McClelland, J. L., and Rumelhart, D. E., 1981, An interactive activation model of context effects in letter perception: Part 1. An account of basic findings. Psychological Review, 88:375-407.

Regier, T., 1992, The acquisition of lexical semantics for spatial terms: A connectionist model of perceptual categorization. PhD Thesis, University of California. Berkeley. Available as TR-92-062, ICSI.

* Shastri, L., 1988, Semantic networks: An evidential formulation and its connectionist realization, Pitman London/Morgan Kaufman Los Altos.

* Shastri, L., and Ajjanagadde V., 1993, From simple associations to systematic reasoning: A connectionist encoding of rules, variables and dynamic bindings using temporal synchrony. Behavioral and Brain Sciences, 16:417-494.

Sun, R., 1992, On variable binding in connectionist networks, Connection Science. 4(2):93-124.

Touretzky, D. S., and Hinton. G. E., 1988, A Distributed Connectionist Production System. Cognitive Science, 12(3):423-466.

Waltz, D. L., and Pollack, J. B., 1985, Massively parallel parsing: a strongly interactive model of natural language interpretation. Cognitive Science, 9:51-74.

FIGURE CAPTIONS

Figure 1. Encoding of predicates, concepts and the rules $\forall x,y,z$ [*give(x,y,z)* $\Rightarrow$ *own(y,z)*], $\forall x,y$ [*buy(x,y)* $\Rightarrow$ *own(x,y)*], and $\forall x,y$ [*own(x,y)* $\Rightarrow$ *can-sell(x,y)*].

Presented at the International Symposium on Dynamics of Neural Systems, Washington DC

CONSTRAINTS ON THE REPRESENTATION, INTEGRATION AND PROCESSING OF DYNAMIC STRUCTURES

Implications Of Using Temporal Synchrony For Encoding Dynamic Bindings[1]

Lokendra Shastri
International Computer Science Institute
1947 Center Street, Suite 600
Berkeley, CA 94704
USA

Introduction: The *dynamic binding problem* is a central problem in neural information processing. During visual processing it takes the familiar form of the segmentation problem and the feature binding problem — visual processing requires the rapid grouping of information over the spatial extent of an object and across different feature maps so that features belonging to one object are grouped together and not confused with those belonging to another. The dynamic binding problem however, is not restricted to vision and arises in any cognitive activity that requires the rapid instantiation and integration of structured representations. In particular, it arises in myriad ways during language understanding where dynamic bindings are required to support among other things, reasoning, syntactic processing, and the integration of syntactic and semantic structures.

A promising solution to the dynamic binding problem based on temporal synchrony has emerged over the past few years. The possibility of using synchronous activity of appropriate cells to encode bindings was suggested several years ago by Malsburg (1981) but the idea has found its full expression more recently with the development of a range of models that use temporal synchrony to carry out segmentation (Malsburg & Buhmann 1992), object recognition (Hummel & Biederman 1992), extraction of attention maps (Nieber, Kock, & Rosin 1993), rapid reasoning (Shastri & Ajjanagadde 1993) and parsing of english (Henderson 1994). Emerging data from neurophysiology also seems to suggest that synchronous activity may play a role in the representation of dynamic bindings in the animal brain (e.g., Eckhorn et al. 1988; Gray & Singer 1989; Kreiter & Singer 1992).

The assumption that dynamic bindings are encoded using synchronous activity has important representational and processing implications. We examine these implications in the context of rapid reasoning and identify several specific constraints on the processing of structured information that are suggested by the use ot synchrony. Work on a model of rapid reasoning and the parsing of english sentences suggests that interesting cognitive tasks can be solved within these constraints. This suggests that synchrony may be a sufficiently powerful mechanism for representing dynamic bindings.

Reasoning and the dynamic binding problem: Although the dynamic binding problem underlies most cognitive tasks, the nature of the representational structures instantiated by dynamic bindings varies from one task to another. Consider the segmentation problem where parts of the input-field have to be attributed to distinct objects. In this task, dynamic bindings may be best viewed as instantiating *sets* — or *unary* relations. All elements in the input-field that belong to the same object are grouped together to form a set. Let us examine the sort of representational structures instantiated by dynamic bindings during rapid (reflexive) reasoning underlying language understanding.[2]

Assume that an agent's long-term memory (LTM) embodies the following systematic knowledge: 'If someone gives a recipient an object then the recipient comes to own that object'. Given the above knowledge an agent would be capable of inferring 'Mary owns a book' on being told 'John gave Mary a book'. A neurally plausible reasoning system should therefore exhibit the following behavior: If the network's pattern of activity

[1]The preparation of this paper was supported by ONR grant N00014-93-1-1149.

[2]Empirical data strongly suggests that inferences required to establish referential and causal coherence occur rapidly and automatically during text understanding (see e.g., Carpenter & Just 1977). Thus certain kinds of inferences can be drawn very rapidly — within a few hundred milliseconds. The speed and spontaneity with which we understand language highlights our ability to perform such inferences without conscious effort — as though they were a *reflexive* response of our cognitive apparatus. In view of this we have described such reasoning as *reflexive*.

is initialized to represent 'John gave Mary a book' then very soon, its activity should evolve to include the representation of 'Mary owns a book'.

A network must solve several technical problems in order to incorporate the above behavior. Before discussing these problems let us introduce some notation. A specific event such as 'John gave Mary a book' can be viewed as an *instance* of the three place relation *give* with roles: *giver*, *recipient*, and *give-object* and expressed as the *fact give(John, Mary, a-book)*. The systematic rule-like knowledge given above about giving and owning may be succinctly expressed as: (1) *give(x,y,z)* $\Rightarrow$ *own(y,z)*, wherein *own* is a two place relation with roles: *owner* and *own-object* and '$\Rightarrow$' informally means 'leads to'.

Dynamic representation of facts requires dynamic bindings: The reasoning system must be capable of rapidly representing facts such as *give(John, Mary, a-book)* as and when they are "communicated" to the system by other perceptual or linguistic processes and as they arise internally as a result of the reasoning process. Note that the fact *give(John, Mary, a-book)* cannot be represented by simply activating the representations of the roles *giver*, *recipient*, and *give-object*, and the constituents 'John', 'Mary', and 'a-book', since such a representation would be indistinguishable from that of *give(a-book, Mary, John)*. This fact, like any other instantiation of an n-ary relation, is a composite structure wherein each constituent fills a distinct role in a relation. Consequently the representation of such as fact requires the representation of appropriate *bindings* between the roles of the relation and its fillers. Thus the dynamic representation of *give(John, Mary, a-book)* requires the creation of dynamic bindings *(giver=John, recipient=Mary, give-object=a-book)*.

The multiple-instantiation problem: Reasoning often requires the simultaneous activation of more than one fact pertaining to the *same* relation. For example, the system may have to encode *give(John, Mary, a-book)* and *give(Mary,Tom,a-car)* at the same time. A reasoning system must be capable of keeping multiple instantiations of the same relation active without cross-talk between instantiations.

Reasoning involves systematic propagation of dynamic bindings: Another problem concerns the dynamic *generation* of inferred facts. For example, starting with a dynamic representation of *give(John, Mary, a-book)* the state of a network encoding rule (1) should evolve rapidly to include the dynamic representation of the inferred fact: *own(Mary, a-book)*. Generating inferred facts involves the systematic *propagation* of dynamic bindings in accordance with various rules embodied in the system. The rule $give(x, y, z) \Rightarrow own(y, z)$ specifies that a *give* event leads to an *own* event wherein the *recipient* of a *give* event corresponds to the *owner* of an *own* event and the *give-object* of a *give* event corresponds to the *own-object* of an *own* event. Thus the application of this rule in conjunction with the instance *give(John, Mary, a-book)* should create an instance of *own* with bindings *(owner=Mary, own-object=a-book)*.

Long-term facts as temporal-pattern matchers: In addition to encoding domain rules, the reasoning system must also be capable of encoding facts in its LTM and using them during recall, recognition, query answering, and reasoning. For example, a reasoning system should be capable of encoding the fact 'John bought a Rolls-Royce' in its LTM and using it to rapidly answer queries such as 'Did John buy a Rolls-Royce?' and 'Does John own a car?' Observe that a fact in LTM should store the associated bindings as *static* bindings within a long-term structure which can interact with dynamic bindings and detect the occurrence of dynamic bindings that match the stored static bindings.

Parsing and the dynamic binding problem: In addition to reasoning, language understanding requires a solution to the dynamic binding problem for syntactic processing and the dynamic linking of syntactic and semantic structures. In particular, parsing requires the extraction of constituents in a sequence of words and determining the appropriate place of these constituents in the overall phrase structure of the sentence. From the point of view of the representation and processing of dynamic bindings one can draw the following analogy between parsing and reasoning: non-terminals of the underlying grammar correspond to 'entities' in the reasoning system, structural relations among non-terminals such as 'dominates' and 'precedes' correspond to 'relations', grammatical constraints and phrase structure combination operators correspond to 'rules', and the representation of the phrase structure during the parsing process corresponds to the collection of 'dynamic facts' active in the network's state of activation.

Overview of a model of reflexive reasoning: We have developed — SHRUTI — a computational model of reflexive reasoning which incorporates solutions to the problems discussed above. SHRUTI can encode a large number of specific facts and general rules involving n-ary relations as well as sub/super ordinate relations between concepts and perform a broad class of reasoning with extreme efficiency. It solves the dynamic binding problem by maintaining and propagating dynamic bindings using synchronous firing of appropriate nodes. The dynamic fact 'John gave Mary a book' is represented in SHRUTI by the clusters for 'John' and 'giver' firing in synchrony; the clusters for 'Mary' and 'recipient' firing in synchrony; and the clusters for an instance of 'book' and 'given-object' firing in synchrony. The view of information processing implied by SHRUTI is one where i) reasoning is the transient but systematic propagation of a *rhythmic* pattern of activity, ii) each entity in the dynamic memory is a phase in the above rhythmic activity, iii) dynamic bindings are represented as the *synchronous* firing of appropriate nodes, iv) rules are interconnection patterns that cause the propagation and transformation of rhythmic patterns of activity, and v) long-term facts are subnetworks that act as temporal pattern matchers and become active when certain cell-clusters fire synchronously.

The details of the model may be found in (Shastri & Ajjanagadde 1993). In brief, we posit that each n-place relation be encoded as a bank of nodes containing n role nodes, a *collector* node and an *enabler* node. Here 'node' refers to an idealized computational device which corresponds to a small cluster of cells. A rule is encoded by linking the roles of the antecedent and consequent relations in accordance with the correspondence between roles specified in the rule. For example, the rule *give(x,y,z)* $\Rightarrow$ *own(y,z)* is encoded by connecting the roles *recipient* and *give-obj* of the relation *give* to the roles *owner* and *own-obj* of the relation *own*, respectively. By virtue of the interconnections between role nodes of *give* and *own*, the state of activation resulting from the pattern of activation for *give(John, Mary, a-book)* leads to a activation pattern wherein the roles *owner* and *own-object* start firing in synchrony with *recipient* and *give-object* respectively, and hence, with *Mary* and *a-book* respectively. Thus starting with a pattern containing the dynamic fact *give(John,Mary,a-book)*, the network state evolves such that the pattern of activation includes the dynamic fact *own(Mary,a-book)*. The key assumption here is that if there is a link from node A to node B, the firing of A will lead to a synchronous firing of B. Note that the time taken to generate a chain of inference is independent of the total number of rules and facts and is just equal to $l * \alpha$ where l equals the *length* of the chain of inference, α equals the time required for connected nodes to synchronize. If we assume a plausible value of α (under 100 milliseconds), SHRUTI demonstrates that it is possible for a system of simple computing elements to encode millions of rules and facts and draw interesting multiple-step inferences within a few hundred milliseconds.

SHRUTI can also encode *long-term* facts and a type hierarchy. The latter allows reference to categories as well as instances in rules, facts, and queries and the encoding of context sensitive rules such as: (*walk-into(x,y)* $\Rightarrow$ *hurt(x)*; but only in the context where the fillers of the two roles of walk-into have the feature solid). In general, the expressive power of SHRUTI is sufficient to encode knowledge structures such as schemas, frames, productions, and if-then rules.

Constraints on reflexive processing predicted by SHRUTI: We describe some of the representational and processing constraints and predictions that follow from our attempt at engineering a reflexive reasoning system based on temporal synchrony. The constraints and predictions relate to (i) the capacity of the 'working memory' underlying reflexive processing, (ii) bounds on the depth of reasoning and differences in the time course of associative priming versus systematic reasoning, (iii) the form of rules that may participate in reflexive processing, and (iv) the need for a large capacity memory capable of storing relation instances in around 1 second.

Working memory underlying reflexive processing: Dynamic bindings, and hence, dynamic (active) facts are represented in SHRUTI as a rhythmic pattern of activity over nodes in the LTM network. In functional terms, this transient state of activation holds information temporarily during an episode of reflexive reasoning and corresponds to the *working memory underlying reflexive reasoning* (WMRR). Note that WMRR is just the state of activity of the LTM network and not a separate buffer. Also note that the dynamic facts represented in the WMRR during an episode of reflexive reasoning should not be confused with the small number of short-term facts an agent may *overtly* keep track of during *reflective* processing and problem solving. WMRR should not be confused with the short-term memory implicated in various memory span tasks (Baddeley 1986). In our view, in addition to the overt working memory, there exist as many "working memories" as their are major processes in the brain.

Our work predicts that the capacity of WMRR is very large but at the same time it is constrained in critical ways. Thus the number of dynamic facts that can *potentially* be present in the working memory at any given time can be as high as $k2 * R$ where $k2$ is the multiple instantiation constant (see below) and R

is the number of relations known to the agent.

Bound on the number of distinct entities referenced in WMRR During an episode of reflexive reasoning, each entity involved in dynamic bindings occupies a distinct phase in the rhythmic pattern of activity. Hence the number of distinct entities that can occur as role-fillers in the dynamic facts represented in the working memory cannot exceed π_{max}/ω where π_{max} is the maximum delay between two consecutive firings of cell-clusters involved in synchronous firing and ω equals the width of the window of synchrony — i.e., the maximum allowable lead/lag between the firing of synchronous cell-clusters. If we assume that neurally plausible value of π_{max} is about 30 ms. and a conservative estimate of ω is around 6 ms. we are lead to the following prediction: As long as the number of distinct entities referenced by the dynamic facts in the working memory is five or less, there will essentially be no cross-talk among the dynamic facts. If more entities occur as role-fillers in dynamic facts, the window of synchrony ω would have to shrink appropriately in order to accommodate all the entities. As ω shrinks, the possibility of cross-talk between dynamic bindings would increase until eventually, the cross-talk would become excessive and disrupt the system's ability to perform systematic reasoning. The exact bound on the number of distinct entities that may fill roles in dynamic facts would depend largest and smallest feasible values of π_{max} and ω respectively. However we can safely predict that the upper bound on the maximum number of entities participating in dynamic bindings can be no more than 10 (perhaps less).

Bound on the multiple instantiation of relations: The capacity of WMRR is also limited by the constraint that each relation can only be instantiated a bounded number of times ($k2$) during an episode of reasoning. In other words the working memory can contain at most $k2$ dynamic facts per relation (we refer to $K2$ as the multiple instantiation constant). Note that the value of $k2$ need not be the same for all relations; some critical relations may have a higher value of $k2$ while some other relations may have a smaller value. The cost of maintaining multiple instantiations turns out to be significant in terms of space and time. For example, the number of nodes required to encode a rule for backward reasoning is proportional to the square of $k2$. Thus a system that can represent three dynamic instantiations of each relation may have up to nine times as many nodes as a system that can only represent one instantiation per relation. Furthermore, the worst case time required for propagating multiple instantiations of a relation also increases by a factor of $k2$. In view of the additional space and time costs associated with multiple instantiation, and given the necessity of keeping these resources within bounds in the context of reflexive processing, we predict that the value of $k2$ is quite small, perhaps no more than 3.

Bound on the depth of the chain of reasoning: Consider the propagation of synchronous activity along a chain of role ensembles during an episode of reflexive reasoning. Two things might happen as activity propagates along the chain of role ensembles. First, the lag in the firing times of successive ensembles may gradually build up due to the propagation delay introduced at each level in the chain. Second, the dispersion within each ensemble may gradually increase due to the variations in the propagation delay of links and the noise inherent in synaptic and neuronal processes. While the increased lag along successive ensembles will lead to a 'phase shift', and hence, binding confusions, the increased dispersion of activity within successive ensembles will lead to a gradual *loss of binding information.* Increased dispersion would mean less phase specificity, and hence, more *uncertainty* about the role's filler. Due to the increase in dispersion along the chain of reasoning, the propagation of activity will correspond less and less to a propagation of role bindings and more and more to an associative spread of activation. For example, the propagation of activity along a chain of rules such as: P1(x,y,z) $\Rightarrow$ P2(x,y,z) $\Rightarrow$ . . . Pn(x,y,z) due to a dynamic fact P1(a,b,c) may lead to a state of activation where all one can say about Pn is this: there is an instance of Pn which involves the entities a, b, and c, but it is not clear which entity fills which role of Pn. In view of the above, it follows that the depth to which an agent may reason during reflexive reasoning is bounded. In other words, an agent may be unable to make a prediction (or answer a query) — even when the prediction (or answer) logically follows from the knowledge encoded in the LTM — if the length of the derivation leading to the prediction (or the answer) exceeds this bound. It should be possible to relate the bound on the depth of reflexive reasoning to specific physiological parameters and pointers to relevant data are welcome.

Form of rules that may participate in reflexive reasoning: Using complexity theory it can be shown that during backward reasoning (i.e., query answering) it is not possible to make use of rules containing equality constraints among antecedent roles unless (i) such roles map to a consequent role in the rule and (ii) the consequent role get bound during the query answering process. A similar constraint applies to forward (predictive) reasoning. These constraints predict that certain queries cannot be answered in a reflexive manner even though the corresponding predictions can be made reflexively. For example, consider an agent whose LTM includes the rule 'if x loves y and y loves z then x is jealous of z', and the long-term facts 'John loves Mary' and 'Mary loves Tom'. We predict that if this agent is asked 'Is John jealous of Tom?', she will be unable to answer the query in a *reflexive* manner. Note that the antecedent of the rule includes the equality condition: the second role of one instance of 'loves' should equal the first role of the other instance of 'love'. Hence, answering this question will require deliberate and conscious processing unless the relevant long-term facts are active in the WMRR for some reason at the time the query is posed. However, an agent who has the above rule about love and jealousy in its LTM would be able infer 'John is jealous of Tom' in a reflexive manner, on being 'told' 'John loves Mary' and 'Mary loves Tom'.

Conclusion: The representational and inferential machinery of SHRUTI is fairly general and can be applied to other problems in cognition that require the expressive power of n-ary relations and depend on the rapid and systematic interaction between long-term and dynamic structures. Thus the constraints and predictions discussed above may carry over to other domains. For example, Henderson (1994) has adopted the SHRUTI model to design a parser of english whose speed is independent of the size of the grammar and that can recover the structure of arbitrary long sentences as long as the dynamic state required to parse the sentence does not exceed the bounds on the parser's working memory. The parser's limited working memory explains a range of linguistic phenomena pertaining to our limited ability to deal with long distance dependencies, local ambiguity, and center-embedding. It lends evidence to our belief that synchrony may eventually turn out to be a sufficiently powerful mechanism for representing dynamic bindings.

References

Baddeley, A. (1986) *Working Memory.* Clarendon Press.

Carpenter, P. A. & Just, M. A. (1977) Reading Comprehension as Eyes See It. In: *Cognitive Processes in Comprehension.* ed. M. A. Just & P. A. Carpenter. Lawrence Erlbaum.

Eckhorn, R., Bauer, R., Jordan, W., Brosch, M., Kruse, W., Munk, M., & Reitboeck, H.J. (1988) Coherent oscillations: A mechanism of feature linking in the visual cortex? Multiple electrode and correlation analysis in the cat. *Biol. Cybernet.* **60** 121-130.

Henderson, J. (1994) *Description based parsing in a connectionist network.* Ph.D. Thesis, Univ. of Pennsylvania, Philadelphia. PA 19104.

Hummel, J. E., & Biederman, I. (1992) Dynamic Binding in a neural network for shape recognition. *Psychological Review* 99:480–517.

Gray, C. M. & Singer, W. (1989) Stimulus-specific neuronal oscillations in orientation specific columns of cat visual cortex. *Proceedings of the National Academy of Science*, Vol. 86, pp. 1698-1702.

Kreiter, A. K. & Singer, W., (1992) Oscillatory Neuronal Responses in the Visual Cortex of the Awake Macaque Monkey. *European Journal of Neuroscience*, Vol. 4, 369-375.

von der Malsburg, C. (1981) The correlation theory of brain function. Internal Report 81-2. Department of Neurobiology, Max-Planck-Institute for Biophysical Chemistry, Gottingen, FRG. 1981

von der Malsburg, C. & J. Buhmann (1992) Sensory segmentation with coupled neural oscillators. Biol. Cybern. 67, 233-242.

Nieber, E., Koch, C. & Rosin, C. (1993) An oscillation-based model for the neuronal basis of attention. *Vision Res.* Vol. 33, No. 18, pp. 2789–2802.

Shastri, L. & Ajjanagadde, V. (1993) From simple associations to systematic reasoning. A connectionist representation of rules, variables and dynamic bindings using temporal synchrony. *Behavioral and Brain Sciences*, 16, 417–494.