Title
A Connectionist Treatment of Negation and Inconsistency

Journal
Proceedings of the Annual Meeting of the Cognitive Science Society, 18(0)

Authors
Shastri, Lokendra
Grannes, Dean J.

Publication Date
1996

Peer reviewed

A Connectionist Treatment of Negation and Inconsistency

Lokendra Shastri and **Dean J. Grannes**
International Computer Science Institute
1947 Center St., Ste. 600
Berkeley, CA 94704
shastri@icsi.berkeley.edu, grannes@icsi.berkeley.edu

Abstract

A connectionist model capable of encoding positive as well as negated knowledge and using such knowledge during rapid reasoning is described. The model explains how an agent can hold inconsistent beliefs in its long-term memory without being "aware" that its beliefs are inconsistent, but detect a contradiction whenever inconsistent beliefs that are within a certain inferential distance of each other become co-active during an episode of reasoning. Thus the model is not logically omniscient, but detects contradictions whenever it tries to use inconsistent knowledge. The model also explains how limited attentional focus or action under time pressure can lead an agent to produce an erroneous response. A biologically significant feature of the model is that it uses only local inhibition to encode negated knowledge. The model encodes and propagates dynamic bindings using temporal synchrony.

Introduction

The ability to perform inferences in order to establish referential and causal coherence and generate expectations plays a crucial role in understanding language (e.g., McKoon & Ratcliff, 1981). Given that we can understand language at the rate of several hundred words per minute, it is also apparent that we can perform the requisite inferences rapidly — as though they were a *reflex* response of our cognitive apparatus. In view of this, we have described such reasoning as *reflexive* (Shastri, 1991).[1] Certain types of negated knowledge also plays a role in such reasoning. If we were told "John has been to Canada" and "John has not been to Europe", we could readily answer the questions (i) "Has John been to North America?", (ii) "Has John been to France?" and (iii) "Has John been to Australia?" with "yes", "no", and "don't know", respectively. We can also reason reflexively with rules involving certain types of negated conditions. So given "John is a bachelor", we can readily answer "no" to "Is John married to Susan?" Observe that answering this question involves the use of negated knowledge that may be approximated as "A bachelor is not married to anyone" (i.e., *bachelor(x)* $\Rightarrow$ $\neg$*married(x,y)*).[2]

Due to the complexity it adds to the inference process, knowledge representation systems often do not deal explicitly with negation. Some models deal partially with negation by adopting the *closed world assumption* in AI. The intuition behind this assumption is as follows: If an agent knows all the relevant facts about some domain, then it may assume that any fact it does not know is false! In view of this assumption, the agent can treat "don't know" answers as "no" answers. The use of the closed world assumption, however, has limited applicability and cannot be a substitute for the ability to explicitly deal with negated information and distinguish between the epistemic states "don't know" and "no".

The encoding of negated knowledge raises the possibility of inconsistencies in an agent's long-term memory (LTM). We often hold inconsistent beliefs in our LTM without being explicitly aware of such inconsistencies. But at the same time, we often recognize contradictions in our beliefs when we try to bring inconsistent knowledge to bear on a particular task. In view of this, a cognitively plausible model of memory and reasoning should allow inconsistent facts and rules to co-exist in its LTM, but it should be capable of detecting contradictions whenever inconsistent beliefs that are within a certain inferential distance of each other become co-active during an episode of reasoning.

Finally, any agent with limited resources must sometimes act with only limited attentional focus and often under time pressure. This means that an agent may sometimes overlook relevant information and act in an erroneous manner. Extended evaluation or an appropriate cue, however, might make the necessary information available and lead to a correct response. Several interesting aspects of such a situation are captured in the following scenario (which we will refer to as the *Post Office Example*):

> John runs into Mary on the street. "Where are you going?" asks John. "To the post office," replies Mary. "But isn't today Presidents' Day?" remarks John. "Oops! I forgot that today was a federal holiday," says Mary after a momentary pause and heads back.

Clearly, Mary had sufficient knowledge to infer that "today" was a postal holiday. But the fact that she was going to the post office indicates that she had assumed that the post office was open. So in a sense, Mary held inconsistent beliefs. John's question served as a trigger and brought the relevant information to the surface and made Mary realize her mistake. A cognitively plausible model should be capable of modeling such situations.

This paper describes a connectionist model that can encode positive as well as negated rules and facts, rapidly perform a class of inferences, and exhibit the desirable properties discussed above. This work extends our work on SHRUTI, a

[1] A formal characterization of reflexive reasoning appears in (Shastri, 1993).

[2] We are using the notation of first-order logic for convenience. This does not mean that we view deduction to be the sole basis of reflexive reasoning. All variables are assumed to be universally quantified.

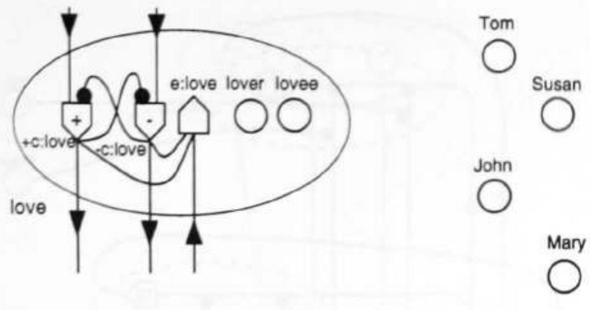

Figure 1: The structure of a predicate cluster.

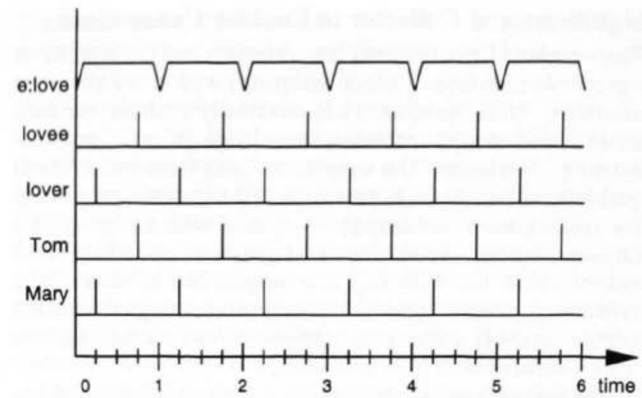

Figure 2: The rhythmic pattern of activation representing the dynamic bindings *love(Mary,Tom)*.

connectionist model of reflexive reasoning (Ajjanagadde & Shastri, 1991; Shastri & Ajjanagadde, 1993; Mani & Shastri 1993) and is partly influenced by (Cottrell, 1985). A detailed description of the extended model appears in (Shastri & Grannes, 1995).

System Overview

This section presents a brief summary of the system. Due to limited space we will not describe the encoding of multiple instantiations, type hierarchy, and context-sensitive rules.

General Representation

Figure 1 illustrates the representation of a predicate and entities. A node such as *John* corresponds to a *focal* node of the representation of the entity "John". Information about the various features of John and the roles he fills in various events is encoded by linking the focal node to appropriate nodes distributed throughout the network (see Shastri & Feldman, 1986; Feldman, 1989).

Encoding of Predicates: Predicate Clusters as Convergence Zones

Consider the encoding of the binary predicate *love* with two roles: *lover* and *lovee*. This predicate is encoded by a cluster of nodes consisting of two role nodes depicted as circular nodes and labeled *lover* and *lovee*; an *enabler* node depicted as a pentagon pointing upwards and labeled *e:love*; and two *collector* nodes depicted as pentagons pointing downwards and labeled *+c:love* and *–c:love* respectively. In general, the cluster for an *n-ary* predicate contains n role nodes, one enabler node, and two collector nodes. The circular nodes are ρ-**btu** nodes while the pentagon shaped nodes are τ-**and** nodes. The computational behavior of these nodes will be described shortly.

The cluster of nodes described above act as an anchor for the complete encoding of a predicate. All rules and facts that involve a predicate converge on its cluster, and all rules and facts involving a predicate can be accessed by fanning out from the predicate's cluster. This representation of a predicate is closely related to the notion of "convergence zones" (Damasio, 1989).

Let us examine the semantic import of the *enabler* and *collector* nodes. Assume that the roles of a predicate P are dynamically bound to some fillers thereby representing a dynamic instance of P (we will see how, shortly). The concomitant activation of the enabler $e{:}P$ means that the system is trying to explain whether the currently active dynamic instance of P is *supported* by the knowledge in the memory. The request for such an explanation might be generated internally by the reasoning system, or be communicated to it by some other subsystem (e.g., the planning module). The semantic import of the two collectors $+c{:}P$ and $-c{:}P$ is the complement of that of the enabler node. The system activates the positive collector $+c{:}P$ when the currently active dynamic instance of P is supported by the knowledge encoded in the system. In contrast, the system activates the negative collector $-c{:}P$ when the *negation* of the active instance is supported by the system's knowledge. Neither collector becomes active if the system does not have sufficient information about the currently active dynamic instance. The collectors can also be used by an external process. For example, the language understanding process might activate *c:love* and establish the bindings *(lover=John, lovee=Mary)* upon hearing the utterance "John loves Mary". Since the two collectors encode mutually contradictory information they have mutually inhibitory links. Observe that this inhibition is *local* to the two collectors within a predicate cluster.

Detecting a Contradiction

The levels of activation of the positive and negative collectors of a predicate measure the effective degree of support offered by the system to the currently active predicate instance. These levels of activation are the result of the activation incident on the collectors from the rest of the network and the mutual inhibition between the two collectors. The two activation levels encode a graded belief ranging continuously from "no" on the one extreme — where only the negative predicate is active, to "yes" on the other — where only the positive collector is active, with "don't know" in between — where neither collector is very active. If both the collectors receive comparable and strong activation then both collectors can be in a high state of activity, in spite of the mutual inhibition between them. When this happens, a contradiction is detected. In the current implementation this is done by an additional node within each predicate cluster (not shown in Figure 1) that has a threshold of 1.5 and receives excitatory inputs from both the collectors.

Significance of Collector to Enabler Connections

The weighted links between the collectors and the enabler of a predicate convert a dynamic assertion into a query about the assertion. Thus the system can constantly evaluate (or seek an explanation for) incoming knowledge in the context of existing knowledge. The weights on links from collectors to enablers can be viewed as a measure of the system's propensity for seeking such evaluations. A system with a high weight on these links can be viewed as a highly critical and skeptical system, while one with very low weights can be viewed as a credulous system — one which accepts incoming information without actively seeking an explanation or determining how well it coheres with prior knowledge.

The system's ability to evaluate incoming information enables it to detect inconsistencies between incoming information and prior knowledge. This evaluation process is fast and automatic but the scope of inconsistency detection is bounded by the constraint on the maximum depth of reflexive reasoning (Shastri & Ajjanagadde, 1993). Observe that here we are referring to a reflexive process of evaluation and not a deliberate search for inconsistencies.

The links from the collectors of a predicate to its enabler also serve to create positive feedback loops of spreading activation and thereby create stable coalitions of active nodes under appropriate circumstances. Assume that the system is seeking an explanation about the currently active instance of P, and therefore, the enabler of P is active. If the memory supports this instance of P it will activate the positive collector of P. This will create a feedback loop — or a stable coalition — consisting of $e{:}P$, the enablers of other predicates participating in the explanation of P, the appropriate collectors of these predicates, $+c{:}P$, and $e{:}P$.

Computational Behavior of Idealized Nodes

If a ρ-btu node A is connected to another ρ-btu node B then the activity of B synchronizes with the activity of A. In particular, a periodic firing of A leads to a periodic and *in-phase* firing of B.

A τ-and node becomes active on receiving a pulse (or a burst of activity) exceeding a minimum duration, π. Thus a τ-and node behaves like a *temporal and* node. On becoming active, it produces an output pulse similar to the input pulse.

A threshold, n (default value 1), associated with a node indicates that the node will fire upon receiving n or more inputs simultaneously (see Shastri & Ajjanagadde, 1993).

Encoding Dynamic Bindings:

Dynamic bindings are represented by the *synchronous* firing of appropriate role and filler nodes. With reference to Figure 1, the *rhythmic* pattern of activity shown in Figure 2 represents the dynamic bindings *(lover=Mary,lovee=Tom)* (i.e., the dynamic fact *love(Mary,Tom)*). Observe that *Mary* and *lover* are firing in synchrony and *Tom* and *lovee* are firing in synchrony. The absolute phase of firing of nodes is not significant. Also since $e{:}love$ is firing, the system is essentially "asking" whether it believes that Mary loves Tom.

As discussed at length in (Shastri & Ajjanagadde, 1993), there exists substantial neurophysiological evidence to suggest that the propagation of synchronous activity is neurally plausible. A detailed review of synchronous cortical activity appears in (Singer, 1993). The idea that synchronous activity can bind features during visual processing had been suggested by von der Malsburg (1986) (also see Bienenstock & Geman, 1995), but SHRUTI is perhaps the first model to demonstrate how synchronous activation can be harnessed to solve complex problems in the representation and processing of conceptual knowledge.

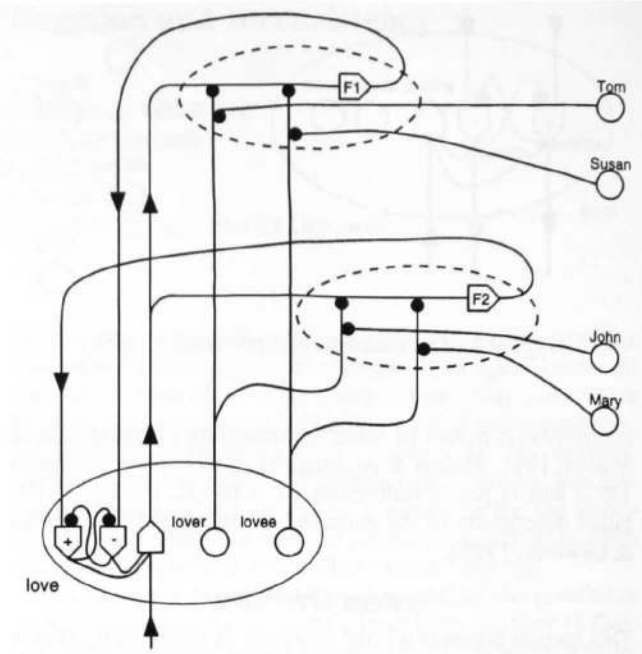

Figure 3: The encoding of facts: *love(John,Mary)* and $\neg$*love(Tom,Susan)*.

Encoding Long-Term Facts: Memory as a Temporal Pattern Matcher

A long-term fact behaves 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. Figure 3 illustrates the encoding of the long-term facts *love(John,Mary)* and $\neg$*love(Tom,Susan)*. Observe that each long-term fact is encoded using a distinct τ-and node which receives a link from the *enabler* node of the associated predicate and sends a link to the positive or negative collector of the predicate depending on whether the fact encodes a positive or a negative fact. The link from the enabler to the fact node is modified by inhibitory links from role nodes of the associated predicate. If a role is bound to an entity, the modifier input from this role node is in turn modified by an inhibitory link from the appropriate entity. Given the query *love(John,Mary)*? the fact node F2 will become active and activate the collector $+c{:}love$ indicating a "yes" answer. Similarly, given the query *love(Tom,Susan)?*, the fact node F1 will become active and activates the $-c{:}P$ collector indicating a "no" answer. Finally, given the query *love(John,Susan)?*, neither $+c{:}love$ nor $-c{:}love$ would become active, indicating that the system can neither affirm nor deny whether John loves Susan.

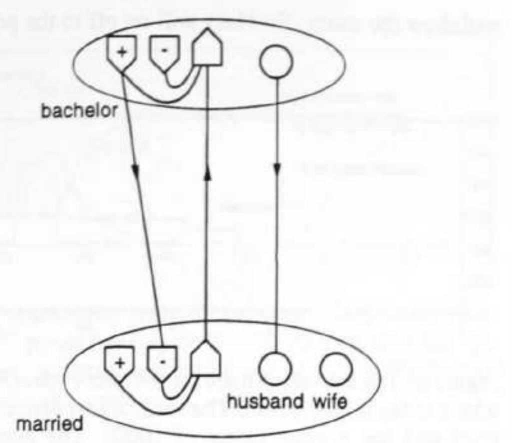

Figure 4: Encoding of the rule: *bachelor(x)* ⇒ *¬married(x,y)*

Encoding of Rules

A rule is encoded by (i) linking the roles of the antecedent and consequent predicates so as to reflect the correspondence between these roles specified by the rule, (ii) connecting the *enabler* of the consequent predicate to the *enabler* of the antecedent predicate, and (iii) connecting the appropriate collectors of the antecedent predicates to the appropriate collector of the consequent predicate. The collector link originates from the positive (negative) collector of an antecedent predicate if the predicate appears in its positive (negated) form in the antecedent. Similarly, the link terminates at the positive (negative) collector of the consequent predicate if the predicate appears in a positive (negated) form in the consequent. Figure 4 shows the encoding of the rule *bachelor(x)* ⇒ *¬married(x,y)*. Observe that the system does not encode the contrapositive of a rule by fiat. In our model, a rule and its contrapositive are two *distinct* rules. Thus the contrapositive form of a rule may, or may not, be present in the LTM.

The encoding of rules makes use of weighted links between predicates. These weights distinguish categorical rules from soft (default) rules and also lead to a gradual weakening of activation along a chain of inference. Eventually the chain of inference terminates when activation falls below a threshold.

The solution to the problem of negation and inconsistency proposed above is simpler than the one suggested in (Cottrell, 1993). The latter suggests duplicating the entire predicate bank for each predicate. In this scheme, each predicate *P* would have two separate banks of role, enabler and collector nodes: one for positive knowledge about *P* (*+P*), and another for negative knowledge about *P* (*–P*). Such a scheme would have required a mechanism for comparing bindings across the *+P* and *–P* banks in order to detect a contradiction.

Three Examples

In this section we present three examples. These have been greatly simplified in order to focus on the key properties of the model.

First, assume that the system has the following rule and fact in its LTM: *bachelor(x)* ⇒ *¬married(x,y)* and *bachelor(John)*. Now the system is told "John is married to Susan" by activating *+c:married* and establishing the dynamic bindings *(husband=John,wife=Susan)*. Activation propagates from *+c:married* to *e:married*, and because of the rule, from *e:married* to *e:bachelor*. The *husband* role of *married* also synchronizes with the role of *bachelor* (refer to Figure 4). At this time, the fact *bachelor(John)* matches the dynamic binding at *bachelor* and activates *+c:bachelor* (the fact is not shown in Figure 4). The activation from *+c:bachelor* propagates down to *–c:married*. Thus both the collectors of *married* become active and signal a contradiction between the agent's existing beliefs and the new information. The system has the option of rejecting the incoming information as spurious or updating its existing beliefs about John. How the system exercises its options is beyond the scope of this work.

Inconsistencies in existing knowledge are also detected in an analogous manner when inconsistent knowledge is activated. This can happen during the processing of a query or during the assimilation of new information. For example, assume that the following (inconsistent) knowledge resides in the LTM:

1. *P(x,y)* ⇒ *R(x,y)*
2. *Q(x,y)* ⇒ *¬R(x,y)*
3. *P(a,b)*
4. *Q(a,b)*

Now assume that the execution of some cognitive task results in the query *R(a,b)?* to the memory and reasoning system. As a result of rules (1) and (2), this query leads to the queries *P(a,b)?* and *Q(a,b)?*. The facts (3) and (4) match the two queries, respectively, and activate *+c:P* and *+c:Q*. These collectors in turn activate *+c:R* and *–c:R* respectively. The activation of the positive and negative collectors of *R* leads to the detection of a contradiction. Thus the proposed encoding allows inconsistent knowledge to reside in the agent's memory, but detects an inconsistency whenever the agent tries to bring some inconsistent knowledge to bear on a particular task.

Next, we describe a simulation of the Post Office Example introduced earlier to illustrate how an agent may overlook relevant information and act in an erroneous manner. Extended evaluation — or an appropriate cue, however, can make the relevant information accessible and lead to the correct response. We model the agent's knowledge as follows (refer to Figure 5):

(i) *presidents-day(day)* ⇒ *federal-holiday(day)*
(ii) *3rd-Mon-Feb(day)* ⇒ *presidents-day(day)*,
(iii) *3rd-Mon-Feb(20-Feb-95)*
(iv) *¬3rd-Mon-Feb(21-Feb-95)*
(v) *weekday(day)* ∧ *post-office(x)* ⇒ *open(x,day)* (with a medium weight)
(vi) *weekend(day)* ∧ *post-office(x)* ⇒ *¬open(x,day)*
(vii) *federal-holiday(day)* ∧ *post-office(x)* ⇒ *¬open(x,day)*
(viii) *post-office(PO)*

The significance of items (i), (v), (vi), and (vii) is fairly obvious. Item (ii) specifies that third Mondays in February are Presidents' Days. Ideally *3rd-Mon-Feb* would be realized as a mental process. We are indirectly simulating such a procedure by assuming that such a mental process is accessed via the predicate *3rd-Mon-Feb* in order to determine whether the day bound to its role is a third Monday in February. In this example, this mental "calendar" consists of two facts stated in items (iii) and (iv). Item (viii) states that PO is a particular post

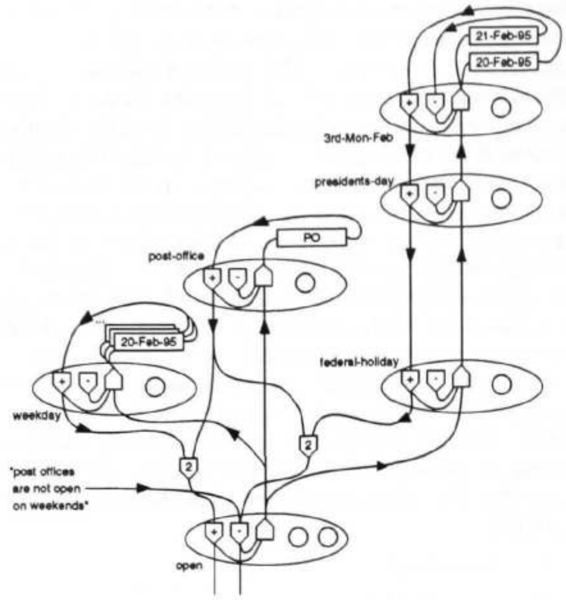

Figure 5: The network representation of the Post Office Example. Links between roles, detailed encoding of facts, the relation *weekend* and the encoding of rule (vi) is not shown. Rules (v) and (vii) are multiple antecedent rules. Thresholds other than 1 are shown inside nodes.

office. Items (i), (ii), (vi), and (vii) are categorical rules about the domain and have a high weight, but item (v) corresponds to default and defeasible information and hence, has a medium weight. In the current implementation, default rules have a weight of 0.70 while categorical rules have a weight of 1. We assume that "Today" is a concept which is bound each day to the appropriate date and to "weekday" or "weekend" depending on the day. These bindings are assumed to be available as facts in the agent's memory.

Imagine it is 20-Feb-95, which is Presidents' Day, and Mary is planning a trip to the post office (PO). Her "go-to-post-office" schema has the precondition that the post office must be open so it poses the query *open(PO,Today)?* Assume that after posing the query the schema monitors the activity of *+c:open* and *–c:open* and accepts an answer based on the criterion: *Accept a "yes" ("no") answer if the positive (negative) collector stays ahead and exceeds a threshold,* θ_{accept}*, for some minimum length of time,* Δ_t. Once the schema accepts an answer, it terminates the query and proceeds with its execution.

Since "Today" is bound to *20-Feb-95*, the fact *weekday(20-Feb-95)* is present in Mary's memory. When the schema asks the query *open(PO,Today)?*, the default rule about post offices remaining open on weekdays becomes active first and activates the positive collector *+c:open* (refer to Figures 5 and 6). If we assume θ_{accept} to be 0.5, the activation of *+c:open* exceeds θ_{accept} after 12 cycles and stays above threshold for about 20 cycles. During this time, the negative collector does not receive any activation and stays at 0. If we assume that Δ_t is 10 cycles, the schema will accept *+c:open* as an answer and withdraw the query. So Mary will set off to the post office.[3]

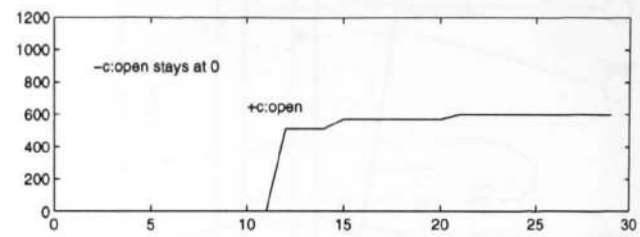

Figure 6: The activation trace for the query *open(PO,Today)?*; where today is 20-Feb-95. The vertical axis denotes activation level and has a scale factor of 1000. The horizontal axis denotes number of simulation steps.

Had the query remained active, the inference process would have eventually inferred that the post office is not open today. The result of the inferential process, if the query *open(PO,Today)?* had not been terminated by the schema, is shown in Figure 7. The dark lines show the activation of the collectors of *open* while the dotted lines show the activation of the collectors of some other relevant predicates. First it is inferred that today is a weekday. Next it is inferred that today is the third Monday in February. As a result, the inference that today is Presidents' Day, and hence, a federal holiday, follows. This in turn leads to the inference that the post office is not open today.

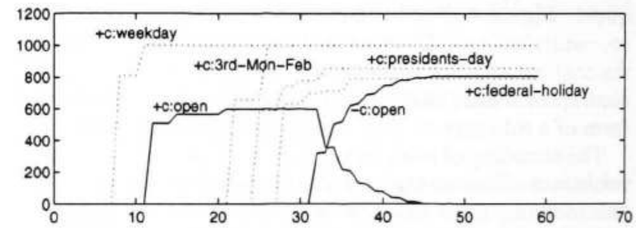

Figure 7: The activation trace for the query *open(PO,Today)?* — today being 20-Feb-95 — allowed to run its full course.

Subsequently, John asks Mary: "Isn't today Presidents' Day?". This causes the language process to activate *e:Presidents-day* and bind the role of Presidents' Day to *20-Feb-95*. This leads to the activation of *e:3rd-Mon-Feb* and then *+c:3rd-Mon-Feb* (via the fact *3rd-Mon-Feb(20-Feb-95)*). The activation from *+c:3rd-Mon-Feb* works its way back and activates *–c:open*. Since this activation is due to categorical rules (rules ii, i, and vii), it is stronger than that arriving at *+c:open* from the default rule (item v). The mutual inhibition between the highly activated *–c:open* and the moderately activated *+c:open* results in the suppression of *+c:open*, making Mary realize that the post office is not open (see Figure 8).

[3] The values of θ_{accept} and Δ_t cited above are the ones used in the simulation.

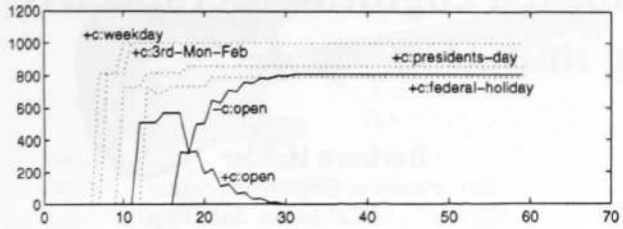

Figure 8: The activation trace for the query "Isn't today Presidents' Day?" posed to Mary on 20-Feb-95 long after her "go-to-post-office" schema has posed the query *open(PO,Today)?* and accepted a yes answer.

Conclusion

This paper describes an extension of the structured connectionist model SHRUTI that can deal with positive as well as negated forms of facts and rules. The model explains how an agent can hold inconsistent beliefs without being "aware" that its beliefs are inconsistent, but detect a contradiction when two contradictory beliefs that are within a small inferential distance of each other become co-active during an episode of reasoning. The model also shows how limited attentional focus or action under time pressure may lead to an erroneous response.

The significance of this work extends beyond reasoning. In essence, SHRUTI demonstrates how connectionist networks can represent relational structures and perform certain types of computations over such structures in an efficient manner. This involves the representation of *static* as well as *dynamic* bindings, interactions between these two types of bindings, and the systematic but context sensitive propagation of dynamic bindings from one relational structure to another. Hence the significance of the representational and inferential mechanisms developed in SHRUTI extends to any cognitive task that involve computations over relational structures such as frames and schemas. For example, Henderson (1994) has shown that the SHRUTI architecture is also appropriate for supporting real-time parsing of English.

In future work we plan a detailed investigation of the interactions between default and categorical rules. In doing so we will draw upon earlier work on connectionist treatment of exceptions, multiple inheritance, and default information (Cottrell, 1985; Shastri, 1988).

Acknowledgment:

This work was partially funded by ONR grants N00014-93-1-1149 and N00014-95-C-0182. Thanks to Jerry Feldman, George Lakoff, D.R. Mani, Srini Narayanan, David Bailey and Dan Jurafsky for intellectual stimulation.

References

Ajjanagadde, V. & Shastri, L. (1991) Rules and variables in neural nets, *Neural Computation*. 3, 121–134.

Bienenstock, E. & Geman, S. (1995) Compositionality in Neural Systems. In M.A. Arbib (Ed.), *The Handbook of Brain Theory and Neural Networks* (pp. 223–226). Cambridge, MA: MIT Press.

Cottrell, G.W. (1985) Parallelism in inheritance hierarchies with exception. In *Proceedings of IJCAI-85* (pp. 194–202). San Mateo, CA: Morgan Kaufmann Publishers.

Cottrell, G.W. (1993) From symbols to neurons: Are we yet there? *Behavioral and Brain Science*, 16:3, 454.

Damasio, A. R. (1989) Time-locked multiregional retroactivation: A systems-level proposal for the neural substrates of recall and recognition. *Cognition*, 33, 25-62.

Feldman, J. A. (1989) Neural Representation of Conceptual Knowledge. In L. Nadel, L.A. Cooper, P. Culicover, & R.M. Harnish (Eds.), *Neural Connections, Mental Computation* (pp. 68–103). Cambridge, MA: MIT Press.

Henderson, J. (1994) Connectionist Syntactic Parsing Using Temporal Variable Binding. *Journal of Psycholinguistic Research*, 23 (5) 353–379.

Mani, D.R. and Shastri, L. (1993) Reflexive Reasoning with Multiple-Instantiation in in a Connectionist Reasoning System with a Typed Hierarchy, *Connection Science*, Vol. 5, No. 3 & 4, 205–242.

McKoon, G., & Ratcliff, R. (1981) The Comprehension Processes and Memory Structures Involved in Instrumental Inference. *Journal of Verbal Learning and Verbal Behavior*, 20, 671-682.

Shastri, L. (1988) *Semantic networks : An evidential formulation and its connectionist realization*. San Mateo, CA: Morgan Kaufmann Publishers.

Shastri, L. (1991) Relevance of Connectionism to AI: A representation and reasoning perspective. In J. Barnden and J. Pollack (Eds.) *Advances in Connectionist and Neural Computation Theory*, vol. 1 (pp. 259–283). Norwood, NJ: Ablex.

Shastri, L. (1993) A Computational Model of Tractable Reasoning – Taking Inspiration from Cognition. In *Proceedings of IJCAI-93* (pp. 202–207). San Mateo, CA: Morgan Kaufmann Publishers.

Shastri, L. & Ajjanagadde, V. (1993) From simple associations to systematic reasoning. *Behavioral and Brain Sciences*, 16:3, 417-494.

Shastri, L. & Feldman, J.A. (1986) Neural Nets, Routines and Semantic Networks. In N. Sharkey (Ed.), *Advances in Cognitive Science* (pp. 158–203). Chichester, UK: Ellis Horwood Publishers.

Shastri, L. & Grannes, D.J. (1995) Dealing with negated knowledge and inconsistency in a neurally motivated model of memory and reflexive reasoning. TR-95-041, ICSI, Berkeley, CA.

Singer, W. (1993) Synchronization of cortical activity and its putative role in information processing and learning. *Annual Review of Physiology* 55: 349-74

von der Malsburg, C. (1986) Am I thinking assemblies? In G. Palm & A. Aertsen (Eds.), *Brain Theory* (pp. 161–176). New York, NY: Springer-Verlag.