A Parallel Natural Language Processing Architecture with Distributed Control

George Berg
Computer Science Department
Northwestern University
Evanston, IL 60201

Abstract

This paper describes work on the autonomous semantic network (ASN) knowledge representation and natural language processing architecture and its implementation - the NO HANS simulator. An ASN is an enhanced spreading activation semantic network, but one without a centralized controller. Rather, in addition to the semantic network are types of nodes which have the ability to change links or add nodes in the network. These nodes are activated by the energy spreading through the underlying semantic network. Thus, the same spreading activation which infuses the knowledge representation also drives the control mechanism as well. Because of this, ASN's offer a compromise between the distributed but restrictive connectionist model and the powerful but heretofore essentially serial conceptual natural language processing models. Spreading activation is also the basis for the search capability, which is loosely based on the connectionist winner-take-all idea. We construct a simple conceptual analyzer in this model and indicate how it works.

1. Introduction

The goal of this work is to be able to use the conceptual style of natural language processing in a massively-parallel form. Conceptual natural language processing merges syntactic, semantic and world knowledge and facilitates processing by allowing high-level inferences (cf Schank and Abelson, 1977; Dyer, 1983). One of the problems with this approach is that the number of alternatives that needs to be searched becomes prohibitive using a serial model of computer control. Massively-parallel architectures (cf Hillis, 1985; Fahlman, 1979) have tens of thousands of nodes, each of which has a limited processing capability. Their forte is extremely fast search. By casting the conceptual techniques into a form where we can utilize parallelism, we hope to gain insights into how to increase the size and complexity of the domains which can be handled by such systems, eventually transcending the micro-world limitiations which have plagued the field.

In recent years there has been much work on distributed natural language processing. These efforts fall into two main categories - connectionist and symbolic. The connectionist systems (Cottrell, 1985; Waltz and Pollack, 1985; Selman and Hirst, 1985) are primarily parsers - they produce a pattern of activation corresponding to a parse tree. They do not add a representation of the sentence to their networks. Because of this, they are not good candidates for conceptual analyzers - an important feature of conceptual systems is that they can build and manipulate a representation of the input text. The symbolic systems, such as the Word Expert Parser (Small and Rieger, 1982) do produce a representation of the new information. Unfortunately, they use a high-level model of parallelism. They use powerful processing units which engage in complex communication with one-another. We would like to be able to use the simpler processing units and communications abilities which are suited to massive-parallelism.

What we propose in this paper is a new model - the autonomous semantic network (ASN). ASN's offer parallelism comparable to the connectionist models while potentially allowing the sophisticated processing of the symbolic, conceptual systems.

2. Autonomous Semantic Networks

The idea underlying an ASN is a spreading-activation semantic network (cf Collins and Loftus, 1975). These systems have used a central, controlling program to direct their activity. Our approach differs because, to take advantage of the parallelism inherent in such an architecture, it is necessary to distribute the controller among nodes connected to the network. To do this, new kinds of nodes were added which alter the network in response to new information. In addition to the normal nodes and links of the spreading activation network, we add *construction nodes* ("c-nodes") which can manipulate the nodes and links. These c-nodes are connected to the semantic network and are enabled by the same activation which spreads through the network. The activation energy which provides focus in the knowledge representation now drives the system's control mechanisms. In this way we distribute the controller. There is also a type of node, the *Winner-take-all* ("wta"), which provides a search ability by comparing the activation of network nodes.

Although the semantic network which underlies this model is essentially connectionist, it was felt that it was necessary to use the c-nodes, with their relatively sophisticated abilities. Current efforts to get connectionist models to do high-level tasks still require large numbers of nodes (cf Touretzky and Hinton, 1985). So, to retain the distributed qualities of connectionism, but to be able to do the relatively sophisticated operations necessary for conceptual natural language processing, the model adds the c-nodes, each with a limited ability to change the network. This way the system is not bound by the computational limits of connectionism, but still retains its amenability to massive-parallelism. For a detailed treatment of the ASN model see Berg (1987).

The behavior of nodes in the spreading activation portion of an ASN is like the units in a local connectionist model (cf Feldman and Ballard,1982). They receive excitatory and inhibitory inputs and, based on a simple activation and decay function, output activation. They are connected by three kinds of links: *activation*, *hierarchical-activation* ("h-act") and *inhibition*. Activation links are one-way connections which spread positive activation between nodes. Inhibition links are one-way connections which spread negative activation.

The h-act links are where we begin to see the interface between the semantic network and the c-nodes. As far as the semantic network is concerned, one of these is a pair of activation links, one in each direction. Their importance, providing a backbone for search, will be discussed in the next section. The input and output to the system will be through the semanitc network. Certain nodes, called *lexical-input* nodes will be activated when the system is to "read" a word. When a node designated as a *lexical-output* node is activated, it prints a word on the system's output. We will identify four types of c-nodes: *link-h-act*, *link-act* , *de-link* and *make*. Link-h-act and link-act nodes will construct a node of the appropriate type between the two nodes to which they point. Since links are directed, these kinds of nodes use two specialized links: *from* and *to* (figure 1). Similarly, a de-link node will remove a link between two nodes. C-nodes can also attach links

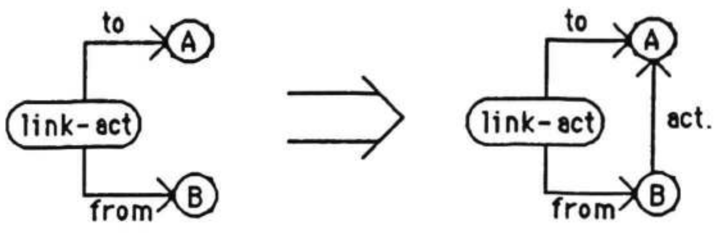

Figure 1. A Link-act node making a link

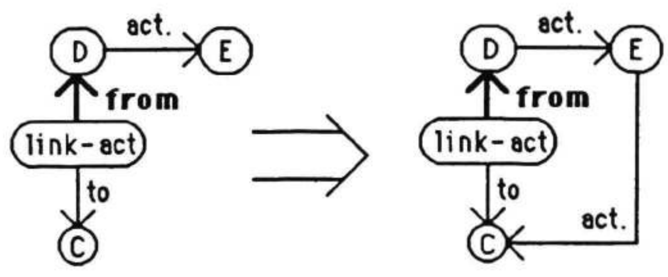

Figure 2. A link-act node using indirection in making a link

indirectly. When a c-node manipulates a link indirectly, it changes the link not on the node to which it points, but to the node to which the second node points (figure 2). Indirect operations by construction nodes will be indicated by bold-face links in the figures. If there is no node connected appropriately to one of its links, a c-node will wait until there is one (however c-nodes, like normal nodes, are subject to becoming inactive through decay). Indirection allows us to use intermediate nodes, or *structural markers*, as pointers to nodes. Structural markers serve an analogous role to slots in a frame system; and the nodes connected to them by activation links can be seen as the slot fillers (see section 4). A make node can introduce new nodes into the network by creating a new node and placing an activation link from itself to the new node.

3. Distributed Search

A conceptual natural language processing system needs a search capability. We provide that by establishing a parallel network which shadows the network of normal nodes and their h-act links. For every normal node which has incoming h-act links there is a *wta* node, and corresponding to every h-act link is a *search* link. Normal nodes which have no incoming h-act links are represented by themselves in this network. The resulting network represents the inheritance hierarchy implicit in the semantic network. The notion of wta comes from Feldman and Ballard's *winner-take-all* networks. When activated, the wta node will select whichever of the nodes connected to it by incoming search links is activated. If a wta node is connected to another by a search link, the subordinate wta node is activated, and the winner of its search is presented to the superordinate wta node. If more than one node is activated, the wta will inhibit all of them and defer a decision until only a single node is active. This way, nodes which will be reactivated by other nodes in the network will be selected in preference to those with spurious or poorly supported activation. In effect, a search starting at a wta node in this network says, "find the

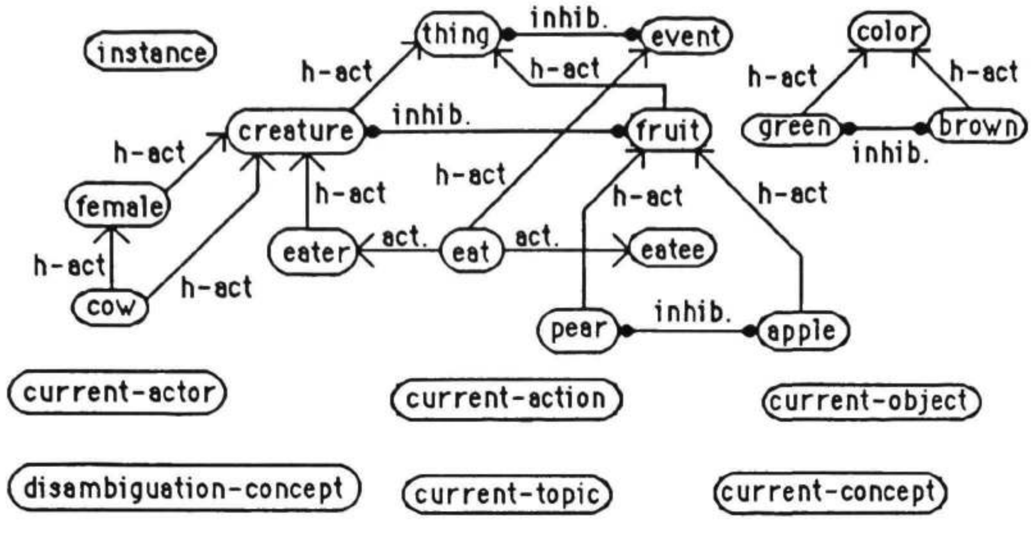

Figure 3. The semantic network before the text is read.

instance (leaf node) in the network below you which is in focus (has the highest supported activation)." The wta node will construct an activation link between itself and the "winner". That way c-nodes can access the winning node indirectly through the wta node.
C-nodes can use their indirection to find the wta node associated with a normal node in the semantic network. This is necessary because, in many cases, the wta node to be activated will not be known *a priori*, and it is necessary to access it through its corresponding normal node. For example if we want to know which instance of "cow" is currently active (or "in focus") we can connect the node for "cow" to a c-node through a structural marker. The c-node will find the wta node associated with the normal node (here **wta-cow**). An example of this is shown in section 4. C-node links which initiate a wta search are identified as *wta-from* and *wta-to* links.

4. Example: A Simple Analyzer

To give an indication of how spreading activation can direct the analysis of a sentence, we give an extremely oversimplified trace of how the sentence "A cow ate an apple" is processed. When a lexical-input node is activated it activates c-nodes which change the network to reflect the word's meaning in the context of the network. Figure 3 shows the semantic network before the sentence is read. To simplify the presentation, the initial network has no nodes which are instances of its concepts and the structural markers are initially unconnected. In this and subsequent figures, h-act links will be shown as one-way nodes to emphasize their role in wta searches, although they spread activation both ways in the semantic network.

Most of the nodes in this figure represent concepts in our taxonomy of things and events. However, we use several structural markers. A structural marker points with an activation link to the node which fills the indicated category. Their purpose is similar to the *binders* of Selman and Hirst (1985) and Cottrell (1985). **Current-concept** is used to indicate which concept is being processed. By connecting **current-concept** to a node with an activation link, the node becomes accesible to c-nodes. In a similar fashion, **current-actor**, **current-action** and **current-object** are used to point to those nodes which represent those categories in our

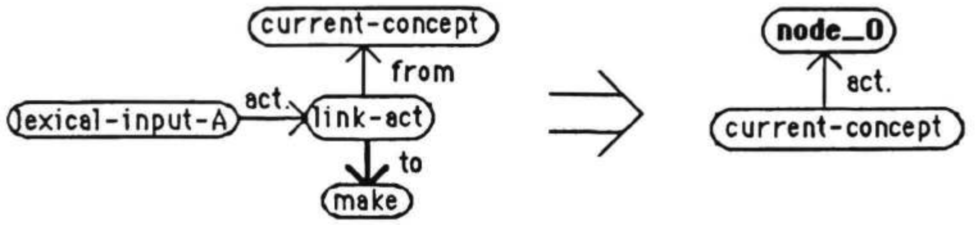

Figure 4. Reading "A"

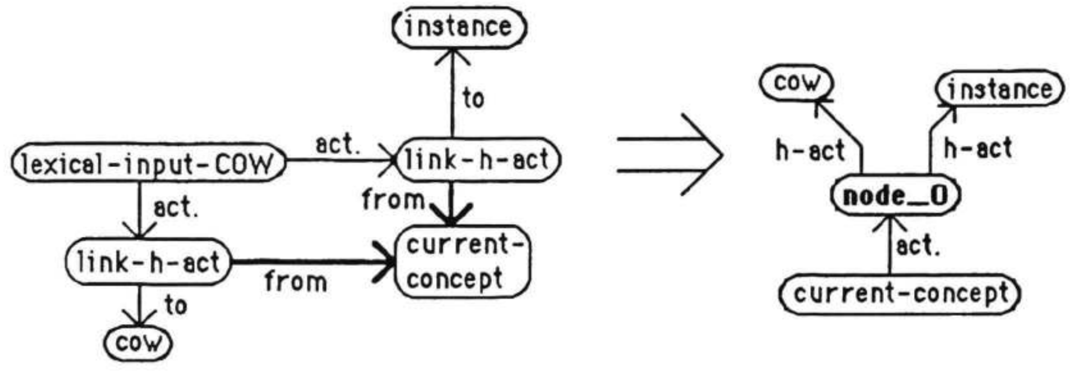

Figure 5. Reading "COW"

sentence. **Disambiguation-concept** is used to point to a node when we want a c-node to access its wta node. **Current-topic** points to a node indicating the overall topic of the current text.

Figure 4 shows the results of activating the lexical node for "A". That node spreads activation to a link-act node. That node constructs an activation link from **current-concept** indirectly through **make**. The indirect reference through **make** causes it to construct a new node, which is the node the link-isa wants. The result of activating those nodes creates a new node (**node_0**) which is the focus of the system's attention by virtue of being connected to **current-concept**.

When **lexical-input-COW** is activated (figure 5) it spreads activation to two link-h-act nodes. When these nodes are activated they connect the node connected to **current-concept** to **instance** and **cow**. In our sentence, **node_0** is identified as an instance of a cow.

When the node for "ATE" is activated, it activates c-nodes which assemble the structure of the sentence (figure 6). **Lexical-input-EAT** (we ignore tense) activates several construction nodes. The three boxes represent groups of c-nodes. The c-nodes in **sub-actor** change **node_0** from having a link from **current-concept** to having one from **current-actor. Sub-action** has c-nodes which make a new node (**node_1**) and link it to **current-action** and establish an activation link from the new node to the current-actor (**node_0**). **Sub-object**'s c-nodes link the node attached to **current-concept** (which is the as yet unread second noun phrase in the sentence - hence figure 6 shows the c-nodes waiting) to **current-object** and creates a link from **node_1** to the node attached to **current-object**. The three link-h-act nodes in figure 6 identify the fillers of the current-actor, current-action and current-object roles as **eater, eat** and **eatee**, respectively, when activated.

Activating the nodes for the second noun phrase ("a pear") works the same as the first. It will

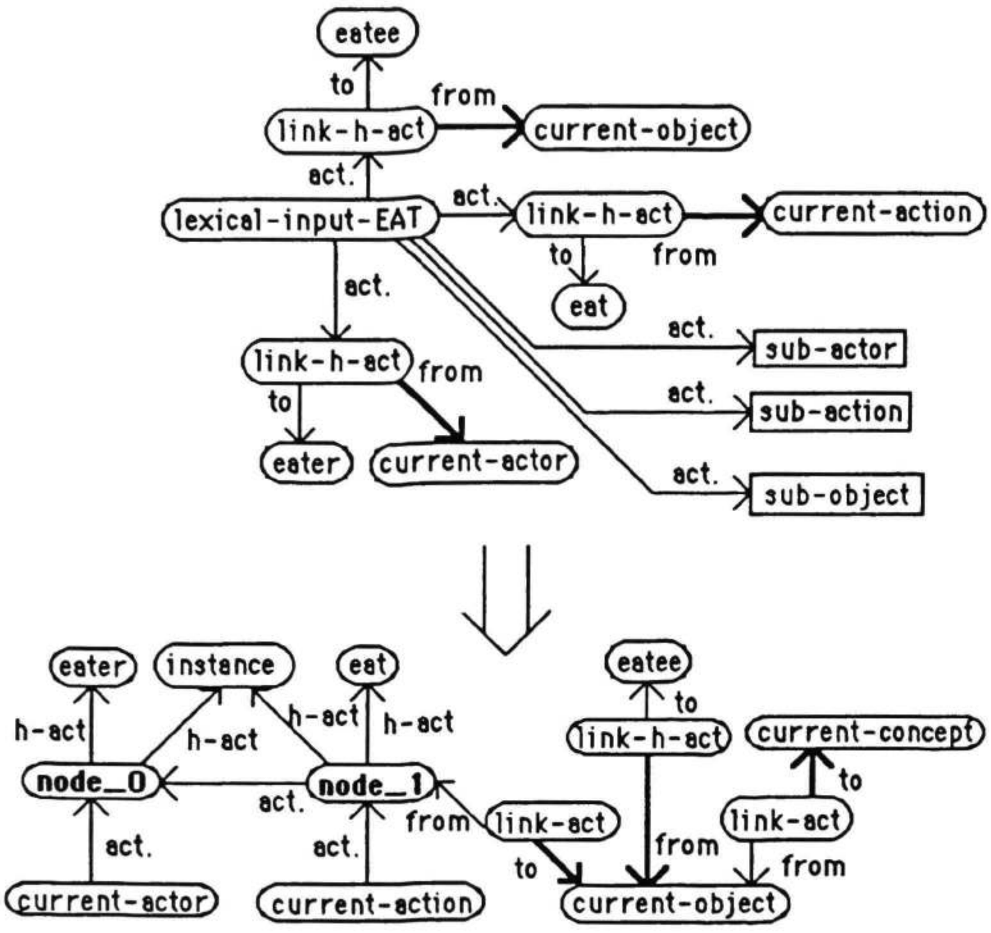

Figure 6. Reading "ATE"

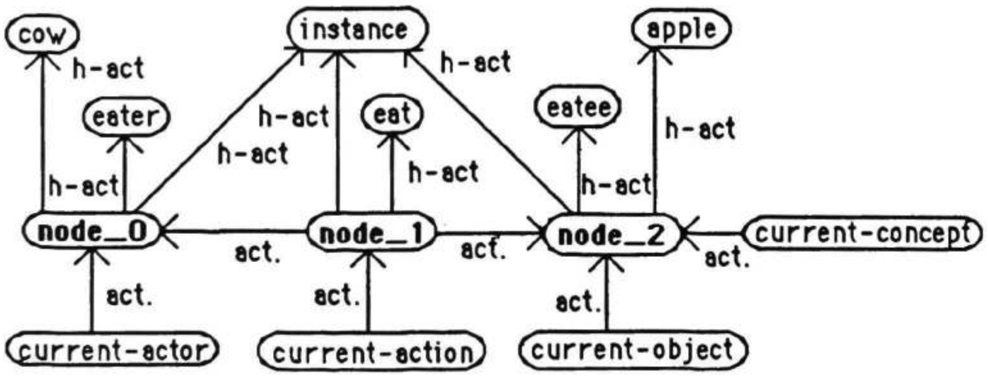

Figure 7. The result of the sentence.

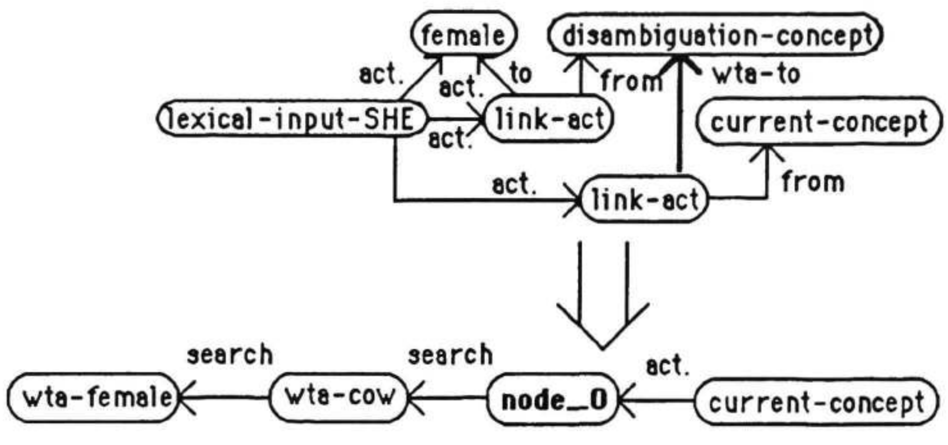

Figure 8. Reading "SHE"

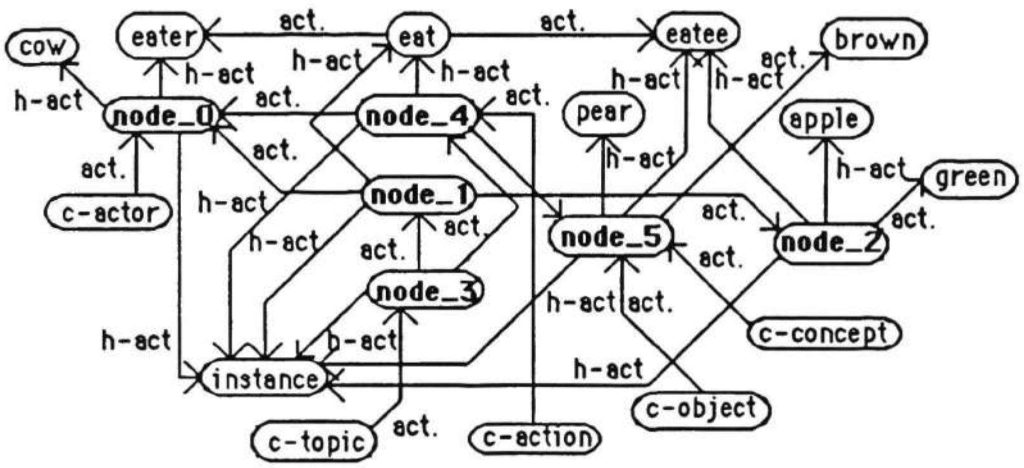

Figure 9. The semantic network after the text has been read.

create a node (**node_2**) and link it to **instance** and **pear. Node_2** will be identified as the current-concept, and thus be acted on by the c-nodes activated by "ATE" and which were stalled waiting for the second noun phrase to be read. Figure 7 shows the semantic network after the entire sentence has been processed.

In addition to our sentence let us consider what the activation of one other lexical-input node does. When the system reads "SHE" (figure 8) two c-nodes are activated. The first links **female** to **disambiguation-concept**. This allows the second to access **wta-female** through an indirect wta-to link. What this does is activate **wta-female.** The wta node will initiate a search. Here its only link is to **wta-cow. Wta-cow** is activated and, in turn does a wta search. In this example, there is only one candidate - **node_0.** Had there been more than one active node along the search paths, they would have been inhibited until only one (presumably reactivated by its neighbors in the semantic network) was active. A pointer to this "winning" node is passed up from **wta-cow** to **wta-female** and is connected by the c-node to the **current-concept**, identifying it as the pronoun's referent.

Our presentation omits many important details of the actual analyzer. It handles definite noun phrases, adjectives and subtle problems with using structural markers which are beyond the scope of this short discussion. To give some flavor of how the system would analyze a text, figure 9 shows the network after it has read "A cow ate an apple. The apple was green. She also ate a brown pear." In addition to the nodes from our sentence, **node_4** represents the "eating event" of the third sentence, **node_5** is the brown pear, and **node_3** is a node representing the entire text.

This analyzer has been implemented on the NO HANS system. NO HANS is a compiled Franz Lisp program which simulates the spreading activation semantic network and the more powerful construction and wta nodes which the network activates. NO HANS supports all of the features of the ASN model, simulating them as a generic, massively-parallel machine. On NO HANS, the analyzer runs the example text in approximately five minutes of real-time on a VAX 780.

6. Future Work

We currently have several directions for our research on ASN's. Our short-term goal is to use them to address many of the issues in conceptual natural language processing. These include handling complex sentence structure, word sense disambiguation, conflict detection, inference and classification. A simple *conflict* would be produced by introducing two contradictory pieces of information such as "Spot is a dog. He is also a pear" or introducing information which violates knowledge already in the network. Our prototypical *inference* problem is "Janet hit Sue. She felt guilty." The inference we need to make is that Janet might feel guilty for hitting Sue, thus being the referent of the pronoun. An example of *classification* would be to have the system read "Clyde is large, has four legs, big ears and a trunk" and have the system's internal representation indicate that Clyde is an elephant.

For the long term, this research is aimed at understanding longer and more complex texts - taking advantage of the distributed control to be able to consider more inferences during processing. As massively-parallel machines come to accomodate this, and more powerful, models, the size and breadth of natural language processing systems can increase. The parallelism will reduce the time necessary to analyze texts. But, ultimately, for any large-scale system of this type to be effective, progress must be made on the ability to make the abstractions and complex inferences necessary for learning. Otherwise we become bogged down creating the representations for every new word and concept our systems will use.

Acknowledgements

I would like to thank Gilbert Krulee, Cathie Love and Betty Modlin, for each helping in their own way. The work presented here benefitted greatly from discussions with John Rager and Jim Roberge´.

References

[1] Berg, George (1987) "Autonomous Semantic Networks for Natural Language Processing', Technical Report in preparation.

[2] Collins, Allan and Loftus, Elizabeth (1975) "A Spreading-Activation Theory of Semantic Processing", *Psychological Review,* 82: 407-428.

[3] Cottrell, Garrison (1985) *A Connectionist Approach to Word Sense Disambiguation,* Ph.D. thesis, University of Rochester, Department of Computer Science, Rochester, NY.

[4] Dyer, Michael (1983) *In Depth Understanding,* MIT Press, Cambridge, MA.

[5] Fahlman, Scott (1979) *NETL: A System for Representing and Using Real-World Knowledge,* MIT Press, Cambridge MA.

[6] Feldman, Jerome and Ballard, Dana (1982) "Connectionist Models and their Properties", *Cognitive Science,* 6: 205-254.

[7] Hillis, W. Daniel (1985) *The Connection Machine,* MIT Press, Cambridge, MA.

[8] Schank, Roger and Abelson, Robert (1977) *Scripts, Plans, Goals and Understanding,* Lawrence Erlbaum, Hillsdale, NJ.

[9] Selman, Bart and Hirst, Graeme (1985) "A Rule-Based Connectionist Parsing System", *Proceedings of the Seventh Annual Conference of the Cognitive Science Society,* Irvine, CA, pp. 212-219.

[10] Small, Steve and Rieger, Chuck (1982) "Parsing and Comprehending with Word Experts", In: Lehnert, W. and Ringle, M. (Eds.), *Strategies for Natural Language Processing,* Lawrence Erlbaum, Hillsdale, NJ.

[11] Touretzky, David and Hinton, Geoffrey (1985) "Symbols among the Neurons: Details of a Connectionist Inference Architecture", *Proceedings of the Ninth International Joint Conference on Artificial Intelligence,* Los Angeles, CA, pp. 238-243.

[12] Waltz, David and Pollack, Jordan (1985) "Massively Parallel Parsing: A Strongly Interactive Model of Natural Language Interpretation", *Cognitive Science,* 9: 51-74.