VITAL

A Connectionist Parser

Tim Howells

Department of Computer and Information Science
University of Massachusetts at Amherst

Abstract

VITAL is an experiment in parsing natural language through the interaction of many local processes without an explicit global control. The global interpretation is achieved through the convergence of a network of processes on a mutually consistent state through cycles of spreading activation and an implicit form of mutual inhibition. This can be viewed as a constraint satisfaction technique consistent with current research both in linguistics and connectionism.[1]

Keywords: Natural language processing, Connectionism

Natural language as constraint satisfaction

In linguistics, computational linguistics and natural language processing the idea is becoming increasingly prevalent that language is better described as the interaction of many weak constraints than as the interpretation of a rigid rule system (Berwick, 1987). Government Binding Theory, Lexical Functional Grammar and Generalized Phrase Structure Grammar all seem to favor a cooperative control structure governed by a multitude of very specific, local constraints distributed throughout the lexicon. Similarly, the Word Expert Parser (Small & Rieger, 1982) has attracted much interest in natural language processing circles as an attempt to define a real formalism based on the sorts of complex lexical interactions that are implicit in the Yale style conceptual analyzers (Birnbaum & Selfridge, 1981). The appeal of this approach is its promise of a more flexible mechanism, with the ability to handle "more or less" grammatical utterances, and to provide reasonable performance in the presence of flaws and shortcomings here and there in the system. It also provides a processing model more consistent with what we know of cognitive processing, especially if the local constraints can be treated as fairly autonomous parallel processes. A problem with the approach is obvious: given a multitude of interacting agents, each with its own local view, how do you control the flow of processing to arrive at a global interpretation at all?

Connectionist models

Connectionist network relaxation models provide an attractive approach to this sort of constraint satisfaction problem (Smolenski, 1986). Communication between processes is limited

[1]This research was supported by an NSF Presidential Young Investigators Award NSFIST-8351863, DARPA contract N00014-87-K-0238, and the Office of Naval Research under a University Research Initiative grant, contract N00014-86-K-0764.

to the passing of numerical activation and inhibition, with the processing state at any point in time defined by the distribution of activation. If all goes well the network will converge on a configuration that represents a maximally consistent global interpretation. This is a truly distributed control structure with a very high degree of parallelism. It also requires that the system be developed according to simple and consistent principles, which may be a benefit in itself.

Gary Cottrell and Steven Small have used this approach to model word sense disambiguation (Cottrell & Small, 1983). Jordan Pollack and David Waltz have proposed it as a way of integrating multiple knowledge sources in natural language processing (Waltz & Pollack, 1985). The systems most similar to my own are the parsers built by Mark Fanty (Fanty, 1985) and Bart Selman (Selman & Hirst, 1985). These both parse according to a phrase structure grammar using spreading activation over a network in which the nodes represent the rules in the grammar. The network converges with a parse tree of the input sentence activated.

VITAL

Distinguishing characteristics

While Mark Fanty used a *dynamic programming* approach and Bart Selman a *Boltzmann machine* approach in their parsers, I've applied an interactive relaxation algorithm similar to that used in McClelland and Rumelhart's word recognition program (1981). I feel that this technique maps more naturally onto the problem, although the network behavior is not as well understood as it is with either of the other two approaches. While Selman and Fanty used prebuilt data structures capable of representing all possible parses of all sentences of up to a fixed length, my networks are built dynamically during parsing. I regard this as a question of implementation with the usual sorts of space/time trade offs.

My relaxation algorithm is unusual in that instead of explicit inhibitory links it utilizes decay over time together with a "competition" for available activation similar to that proposed by James Reggia (1987). This produced a number of nice results which will be detailed below.

The background grammar

The phrase structure grammar is encoded as a collection of "template" network nodes, each of which represents a single rule. The essential information in these template nodes is:

- The rule itself; *e.g.* **NP** $\Rightarrow$ **DET** $\overline{\mathbf{N}}$ [2]
- Pointers to other rules (I'll call them *context rules*) that have the represented rule's left-hand-side as a constituent of their right-hand-sides. Two context rules for the above rule would be:
  1. **S** $\Rightarrow$ **NP VP**
  2. **PrepPhr** $\Rightarrow$ **Prep NP**
- Weights reflecting the statistical likelihood that the represented rule is in fact a constituent of each of those context rules, assuming that the represented rule is in the parse tree. These are simply counters that are incremented every time the two rules participate together in a successful parse.

[2] $\overline{\mathrm{N}}$ (N bar) is an intermediate stage in forming a noun phrase that may be completed by tacking on a determiner. It may be a single noun or a complex phrase like "woman that we saw on the way to the park".

Table 1: A sample grammar fragment

syntactic unit	*context rules*	*count*
book	N ⇒ book	18
	V ⇒ book	3
DET	NP ⇒ DET $\overline{\text{N}}$	208
N	$\overline{\text{N}}$ ⇒ N	172
$\overline{\text{N}}$	NP ⇒ DET $\overline{\text{N}}$	208
	$\overline{\text{N}}$ ⇒ ADJ $\overline{\text{N}}$	35
the	DET ⇒ the	72

As needed during parsing, copies of these "templates" are instantiated as nodes in the network.

A distinction is made between "branching" rules that have more than one constituent on their right hand sides, and "non-branching" rules which have only a single constituent. There is a qualitative difference between these two classes of rules: a branching rule specifies a way in which its constituents may be combined, while a non-branching rule merely provides another name by which to refer to its constituent.

A portion of the grammar needed for the parsing of noun phrases might be recorded as shown in the above table. In this example, the parser has currently seen a total of 243 instances where an $\overline{\text{N}}$ node was incorporated into a parse tree. In 208 of those cases it was simply joined with a determiner to form a noun phrase (NP) as in "the house". In the remaining 35 cases where an $\overline{\text{N}}$ node occurred it was combined with an adjective to form a new $\overline{\text{N}}$ such as "red house". The rules shown here are oversimplified for the sake of an easy example.

A general description of the algorithm

Network building

Network building begins with the instantiation of the words of the input sentence as the terminal nodes. These are given an activation of 200 which remains constant throughout the parse. When a node comes into existence it first activates all its non-branching context rules. These are activated immediately because they merely represent alternative ways to refer to the original node and not a further hypothesis regarding how other consitituents might be combined. As nodes in the network reach a threshold activation level (currently set at 110) they activate new nodes corresponding to their branching context rules. The new nodes are initialized with an activation level proportional to the activation levels of their constituents and the counters associated with its context rule. I'll give an example of this below. When a node reaches an activation level of zero it is removed from the network.

The process of network building is illustrated in *figure 1*, which shows three steps in parsing the noun phrase "the book" using the grammar of *table 1*.

(a) The input nodes are instantiated along with their non-branching context-nodes (and their non-branching context nodes' non-branching context-nodes, *etc.*).

(b) As existing nodes pass the threshold activation level they activate new nodes representing their context-nodes. These new nodes look on lists of existing nodes to find their other constituents. Internal bookeeping prevents the creation of duplicate nodes.

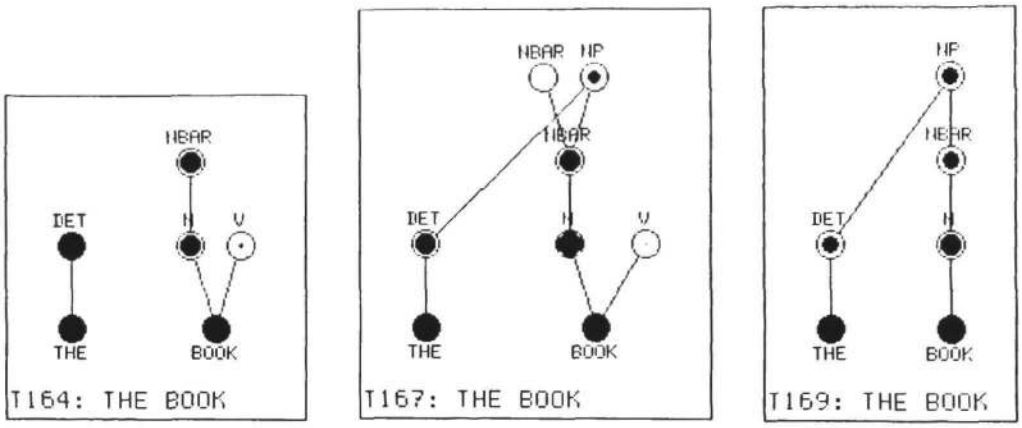

Figure 1: parsing "the book"

(c) During the network relaxation superfluous nodes reach zero activation levels and are removed from the network.

To give an example of how activation levels are initialized, when the "book" node activated its context rules "N" and "V" it had an activation level of 200; the frequency counts it had were 208 for "N" and 35 for "V" (totaling 243); therefore it created an "N" hypothesis with an activation of $200 \times \frac{208}{243}$ or 171, and a "V" hypothesis with an activation of $200 \times \frac{35}{243}$ or 29. As nodes aquire new constituents they immediately get more activation from them according to the same formula. The effect of this procedure with realistic grammars is that syntactic units like $\overline{\text{N}}$ that participate in many different constructions will activate many relatively weak hypothesis, while the function words (*e.g.* articles and prepostitions) will activate only a small number of very strong hypotheses.

Network relaxation

With each cycle each node sends some portion of its total activation to its connected nodes and then the new activation levels are subject to decay. The terminal nodes representing the input send activation but do not receive it and are not subject to decay. There are distinct procedures for sending activation to the nodes that want the sending node as a constituent (bottom-up activation) and the nodes that *it* wants as constituents (top-down activation):

- *Bottom-up activation:* The total amount of activation to be divided among the nodes competing for the sending node as a constituent is calculated as a percentage of its activation. Then it's apportioned according to the square of their current activation levels, so that nodes that already have high activation levels attract large amounts of activation from their constituents, while those with low activation attract little. Say the system parameter controling bottom-up activation is P (I'm currently setting P at 0.4). If a node with an activation level of A is sending bottom-up activation to n other nodes and the activation of the i^{th} of those n nodes is a_i. Then the activation given to the i^{th} node will be:

$$A \times P \times \frac{a_i^2}{\sum_{j=1}^{n} a_j^2}$$

- *Top-down activation:* An amount of activation, again equal to a percentage (currently 40%) of the sending node's activation, is divided equally among the sending node's constituents.

These procedures reflect the fact that each node in the final parse should have *all* its available constituent nodes, but should only be a constituent of a single node, hence the winner take all situation for bottom-up activation.

After each cycle of bottom-up and top-down activation the nodes are subject to a decay rate controled by another system parameter (currently set at 0.8). A penalty for nodes that do not have all of their consituents is factored in. If the parameter is P, a node's activation before decay is A, the number of constituents it requires is C_r and the number of constituents it actually has is C_a then after decay its activation is:

$$A \times (1 - P) \times \frac{C_a}{C_r}$$

An example

Figures two through six show the network at various stages while parsing the sentence "Mary gave the woman wearing a yellow dress the book." The parse is complicated by the fact that the word "yellow" is not in the lexicon. *VITAL* uses top-down expectations to complete the pattern. The dotted lines in the network graphs indicate the "guesses" it made. In this example the parse tree was derived in 24 cycles and the network converged in 34 cycles.

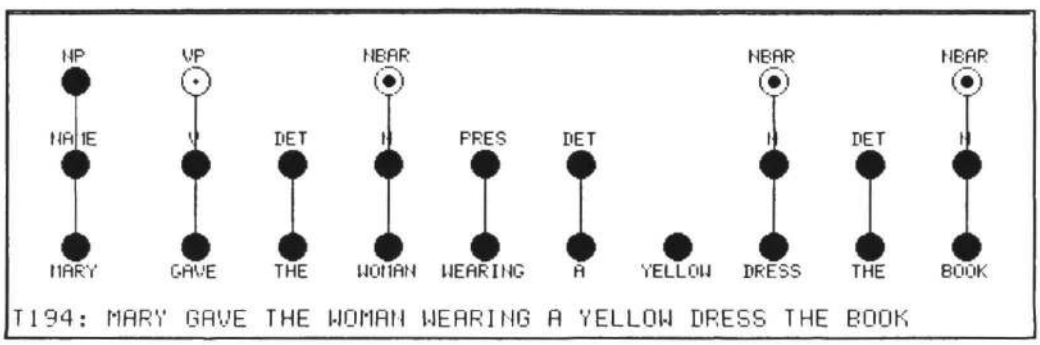

Figure 2: Instantiation of the input nodes

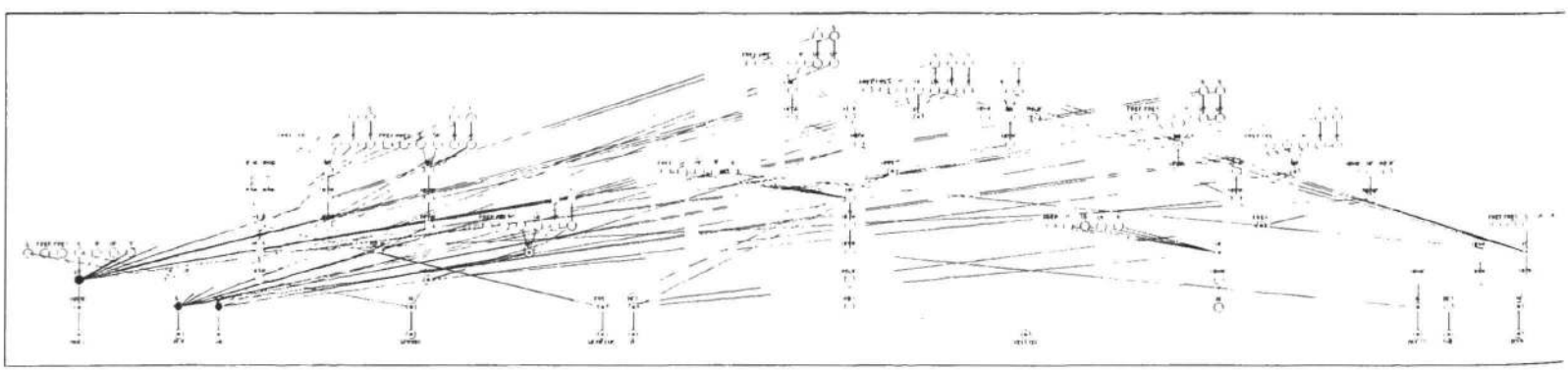

Figure 3: The network after 1 cycle

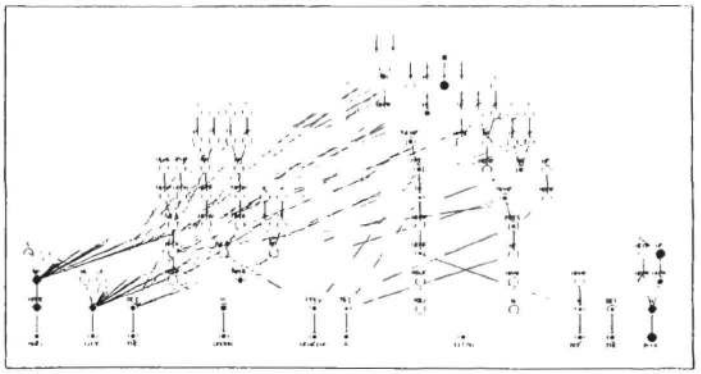

Figure 4: ...after 5 cycles

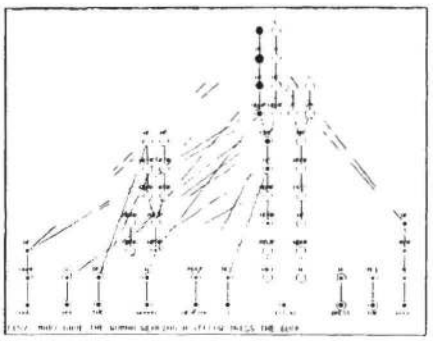

Figure 5: ...15 cycles

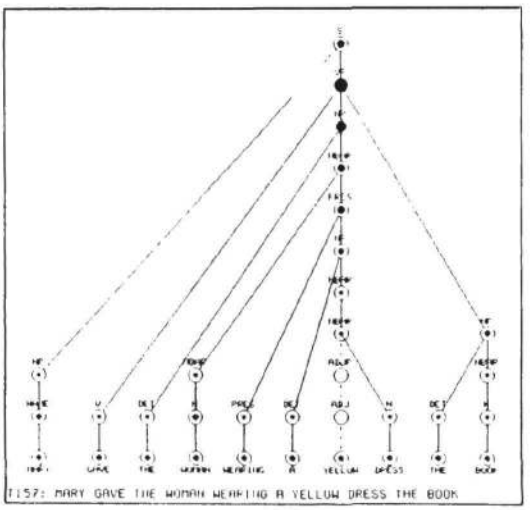

Figure 6: The network after 25 cycles

Discussion

A nice feature of this algorithm is a smooth integration of bottom-up and top-down processing. In the early cycles the activation is strongest in the nodes at the bottom of the network. After several cycles the activation is distributed more evenly, and in the late cycles the heaviest concentration of activation is in a few nodes near the top of the network. The more usual technique of using explicit inhibitory links requires the use of a squashing function to keep the activation levels within preset bounds. By using dynamically changing weights and decay, the nodes converge on values reflecting the amount of activation available to them across the network. Island driving effects arise naturally, since nodes representing hypotheses with a complex and unambiguous internal structure attract large amounts of activation relative to simpler or confused hypotheses. The use of inhibitory links also entails a much more highly interconnected network (Reggia, 1987).

Hinton and Sejnowski have described a major concern regarding relaxation algorithms:

> We would like these networks to settle into states in which a few units are fully active and the rest are inactive. Such states constitute clean "digital" interpretations. To prevent the network from hedging its bets by settling into a state where many units are slightly active, it is usually necessary to use a strongly nonlinear decision rule, and this also speeds up convergence. (Hinton and Sejnowski, 1986)

With my algorithm I found that I could get the desired "clean interpretations" with real-valued units by allowing a very active flow of energy through the system. That is to say the network would settle with all nodes either highly activated or completely off if I set the system parameters so that the nodes send large amounts of activation and are subject to a high rate of decay. [3]

A good test of this behavior is highly ambiguous input where you might expect the activation to get distributed among the equally valid interpretations rather than settling decisively on one of them. I tested this with the noun phrase "the green green dress wearing woman" which has three equally valid syntactic interpretations. *VITAL* generates all three interpretations in its network but settles cleanly on a single one. This is a particularly important quality in natural language processing, since a typical sentence in a business letter can have a few hundred syntactic interpretations. It would be nice to be able to bias the network in favor one or another interpretation, but in order to do this effectively I'll have to introduce some additional sophistication in the network nodes.

I think that this network relaxation algorithm is an intuitively pleasing style of parsing. As described above, a simple distributed control structure allows bottom-up and top-down effects to be integrated smoothly and effectively, and the procedure for initializing the activation levels of new nodes based on the local statistical memory of the existing nodes causes function words to dominate processing in a natural way. This memory based approach also leaves room for learning over time, although this is not a learning program in the usual connectionist sense. The next natural step with this parser would be to incorporate a semantic component that does limited inference using the same constraint satisfaction approach with local associative memories. There are formidable problems in trying to extend this approach to semantic processing, however, mainly because we can't represent semantic structures as cleanly as the syntactic structures of a context-free grammar.

Acknowledgements

Thanks to Wendy Lehnert for her support and for introducing me to these ideas with her *ELAN* program. Also to Steve Bradtke for many useful discussions and criticisms, and for his graphics package, without which this might never have gotten off the ground.

References

Birnbaum, L. and M. Selfridge. 1981. "Conceptual analysis of natural language". in *Inside Computer Understanding.* ed. Schank and Riesbeck. Hillsdale, NJ: Lawrence Erlbaum

Berwick, R.C.. 1987. "Principle based parsing". technical report AI-TR-972, Artificial Intelligence Laboratory, The Massachusetts Institute of Technology, Cambridge, MA. to appear in *The Processing of Linguistic Structure.* Cambridge: MIT Press

Cottrell, G.W. and S.L. Small. 1983. "A Connectionist Scheme for Modelling Word-Sense Disambiguation". *Cognition and Brain Theory* 6, 89-120

Fanty, M.. 1985. "Context-free parsing in connectionist networks". TR-174. Computer Science Department, the University of Rochester, Rochester, NY 14627

[3] I currently am having the nodes send out 80% of their activation levels to their connected nodes and am subjecting them to an 80% decay rate with each cycle.

Hinton, G.E. and T.J. Sejnowski. 1986. "Learning and relearning in Boltzmann Machines". in Rumelhart, McClelland *et al.*, 1986

Lehnert, W.G.. 1987. "Learning to integrate syntax and semantics". *Proceedings of the Fourth International Conference on Machine Learning.* Irvine, CA

McClelland, J.L. and D.E. Rumelhart. 1981. "An interactive activation model of context effects in letter perception: Part 1. An account of basic findings". *Psychological Review*, 88, 375-407

Reggia, J.A.. 1987. "Properties of a competition-based activation mechanism in neuromimetic network models". *Proceedings of the IEEE First International Conference on Neural Networks*, II,131.. San Diego, CA

Rumelhart, D.E., J.L. McClelland *et al.*. 1986. *Parallel Distributed Processing; vol. 1.* Cambridge: MIT Press

Selman, B. and G. Hirst. 1985. "A rule based connectionist parsing system". *Proceedings of the Seventh Annual Conference of the Cognitive Science Society.* Irvine, CA, 212-221

Small, S. and C. Rieger. 1982. "Parsing and comprehending with word experts (a theory and its realization)". in *Strategies for Natural Language Processing*, ed. Lehnert and Ringle. Hillsdale, NJ: Lawrence Erlbaum

Smolenski, Paul. 1986. "Information processing in dynamical systems: Foundations of harmony theory". in Rumelhart, McClelland *et al.*, 1986

Waltz, D.L. and J.B. Pollack. 1985. "Massively parallel parsing". *Cognitive Science*, 9, 51-74