MOVEMENT IN ACTIVE PRODUCTION NETWORKS

Mark A. Jones
Alan S. Driscoll

AT&T Bell Laboratories
Murray Hill, New Jersey 07974

ABSTRACT

We describe how movement is handled in a class of computational devices called *active production networks* (*APNs*). The APN model is a parallel, activation-based framework that has been applied to other aspects of natural language processing. The model is briefly defined, the notation and mechanism for movement is explained, and then several examples are given which illustrate how various conditions on movement can naturally be explained in terms of limitations of the APN device.

1. INTRODUCTION

Movement is an important phenomenon in natural languages. Recently, proposals such as Gazdar's derived rules (Gazdar, 1982) and Pereira's extraposition grammars (Pereira, 1983) have attempted to find minimal extensions to the context-free framework that would allow the description of movement. In this paper, we describe a class of computational devices for natural language processing, called *active production networks* (*APNs*), and explore how certain kinds of movement are handled. In particular, we are concerned with left extraposition, such as Subject-auxiliary Inversion, *Wh*-movement, and NP holes in relative clauses. In these cases, the extraposed constituent leaves a trace which is inserted at a later point in the processing. This paper builds on the research reported in Jones (1983) and Jones (forthcoming).

2. ACTIVE PRODUCTION NETWORKS

2.1 The Device

Our contention is that only a class of parallel devices will prove to be powerful enough to allow broad contextual priming, to pursue alternative hypotheses, and to explain the paradox that the performance of a sequential system often degrades with new knowledge, whereas human performance usually improves with learning and experience.[1] There are a number of new parallel processing (connectionist) models which are sympathetic to this view—Anderson (1983), Feldman and Ballard (1982), Waltz and Pollack (1985), McClelland and Rumelhart (1981, 1982), and Fahlman, Hinton and Sejnowski (1983).

Many of the connectionist models use iterative relaxation techniques with networks containing excitatory and inhibitory links. They have primarily been used as best-fit categorizers in large recognition spaces, and it is not yet clear how they will implement the rule-governed behavior of parsers or problem solvers. Rule-based systems need a strong notion of an operating state, and they depend heavily on appropriate variable binding schemes for operations such as matching (e.g., unification) and recursion. The APN model directly supports a rule-based interpretation, while retaining much of the general flavor of connectionism. An active production network is a rule-oriented, distributed processing system based on the following principles:

1. Each node in the network executes a uniform activation algorithm and assumes states in response to messages (such as expectation, inhibition, and activation) that arrive locally; the node can, in turn, relay messages, initiate messages, and spawn new instances to process message activity. Although the patterns that define a node's behavior may be quite idiosyncratic or specialized, the algorithm that interprets the pattern is the same for each node in the network.

2. Messages are relatively simple. They have an associated time, strength, and purpose (e.g., to post an expectation). They do *not* encode complex structures such as entire binding lists, parse trees, feature lists, or meaning representations.[2] Consequently, no structure is explicitly built; the "result" of a computation consists entirely of the activation trace and the new state of the network.

Figure 1 gives an artificial, but comprehensive example of an APN grammar in graphical form. The grammar generates the strings—*a, b, acd, ace, bcd, bce, fg* and *gf*—and illustrates many of the pattern language features and grammar writing paradigms. The network responds to *sources* which activate the network at its leaves. Activation messages spread "upward" through the network. At conjunctive nodes (*seq* and *and*), expectation messages are posted for the legal continuations of the pattern; inhibition messages are sent down previous links when new activations are recorded.

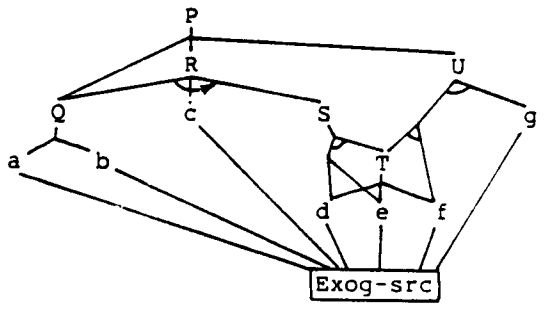


Figure 1. A Sample APN

In parsing applications, partially instantiated nodes are viewed as phrase structure rules whose next constituent is expected. The sources primarily arise from exogenous

1. The human ability to perform computationally expensive operations using relatively slow, parallel hardware reinforces this belief.

2. For a similar connectionist view, see Feldman and Ballard (1982) or Waltz and Pollack (1985). A comparison of marker passing, value passing and unrestricted message passing systems is given in Fahlman, Hinton and Sejnowski (1983).

strobings of the network by external inputs. In generation or problem solving applications, partially instantiated nodes are viewed as partially satisfied goals which have outstanding subgoals whose solutions are desired. The sources in this case are endogenously generated. The compatibility of these two views not only allows the same network to be used for both parsing and generation, but also permits processes to share in the interaction of internal and external sources of information. This compatibility, somewhat surprisingly, turned out to be crucial to our treatment of movement, but it is also clearly desirable for other aspects of natural language processing in which parsing and problem solving interact (e.g., reference resolution and inference).

2.2 The Pattern Language

Each node in an APN is defined by a pattern, written in the pattern language of Figure 2. A pattern describes the messages to which a node responds, and the new messages and internal states that are produced. Each subpattern of the form (*$ v binding-pat*) in the pattern for node *N* is a variable binding site; a variable binding takes place when an instance of a node in *binding-pat* activates a reference to variable *v* of node *N*. Implicitly, a pattern defines the set of states and state transitions for a node. The ? (optionality), + (repetition) and * (optional repetition) operators do not extend the expressiveness of the language, but have been added for convenience. They can be replaced in preprocessing by equivalent expressions.[3] Formal semantic definitions of the message passing behavior for each primitive operator have been specified.

pattern ::= *binding-site*
| (seq *pattern* ...)
| (and *pattern* ...)
| (or *pattern* ...)
| (? *pattern*)
| (+ *binding-site*)
| (* *binding-site*)

binding-site ::= (*$ var binding-pattern*)

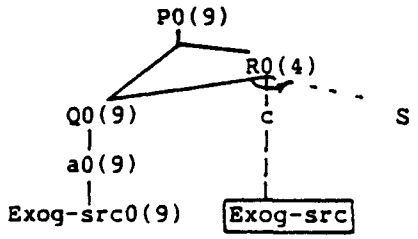
binding-pattern ::= *node*
| (and *binding-pattern* ...)
| (or *binding-pattern* ...)

Figure 2. The APN Pattern Language

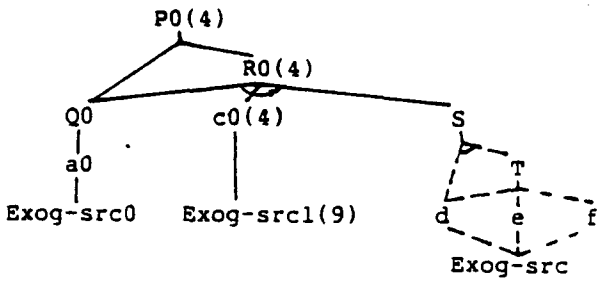
An important distinction that the pattern language makes is in the synchronicity[4] of activation signals. The pattern (and (*$ v1 X*) (*$ v2 Y*)) requires that the activation from *X* and *Y* emanate from distinct network sources, while the pattern (*$ v* (and *X Y*)) insists that instances of *X* and *Y* are activated from the same source. In the graphical representation of an APN, synchrony is indicated by a short tail above the subpattern expression; the definition of *U* in Figure 1 illustrates both conventions: (and (*$ v1* (and *T f*)) (*$ v2 g*)).

2.3 An Example

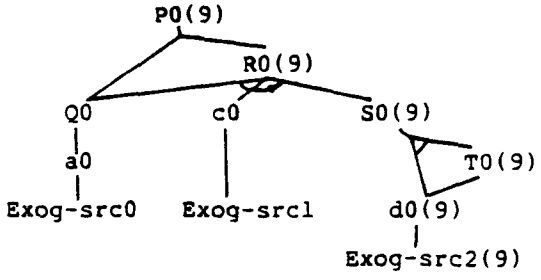
Figure 3 shows the stages in parsing the string *acd*. An exogenous source *Exog-src0* first activates *a*, which is not currently supported by a source and, hence, is in an inactive state. The activation of an inactive or inhibited node gives rise to a new instance (*a0*) to record the binding. The instance is effectively a new node in the network, and derives its pattern from the spawning node. The activation spreads upward to the other instances shown in Figure 3(a). The labels on each node indicate the current activation level, represented as an integer between 0 and 9, inclusive.



(a) trace structure after *a*



(b) trace structure after *ac*



(c) trace structure after *acd*

Figure 3. Stages in Parsing *acd*

3. The exact choice of operators in the pattern language is a somewhat separate issue from the specification of the APN machine.

4. The current APN model allocates sources sequentially. The term *synchronicity* reflects the fact that the source identity of two activation messages can be locally computed from their time of arrival. This works as long as the activation process runs fast enough to condition the network between serial source activations. Alternatively, activation messages could carry the source identity as an additional parameter; in this case, source activations can overlap, but at the possible risk of an inconsistent expectation environment. For relatively independent tasks, the overlap may not pose a problem.

The activation of a node causes its pattern to be (re)instantiated and a variable to be (re)bound. For example, in the activation of *R0*, the pattern (seq ($ *v1 Q*) ($ *v2 C*)) is replaced by (seq ($ *v1* (or *Q Q0*)) ($ *v2 C*)), and the variable *v1* is bound to *Q0*. For simplicity, only the active links are shown in Figure 3. *R0* posts an expectation message for node C which can further its pattern. The source *Exog-src0* is said to be *supporting* the activation of nodes *a0, Q0, R0 and P0* above it, and the expectations or inhibitions that are generated by these nodes. For the current paper we will assume that exogenous sources remain fully on for the duration of the sentence.[5]

In Figure 3(b), another exogenous source *Exog-src1* activates *c*, which furthers the pattern for *R0*. *R0* sends an inhibition message to *Q0*, posts expectations for *S*, and relays an activation message to *P0*, which rebinds its variable to *R0* and assumes a new activation value. Figure 3(c) shows the final situation after *d* has been activated. The synchronous conjunction of *S0* is satisfied by *T0* and *d0*. *R0* is fully satisfied (activation value of 9), and *P0* is re-satisfied.

2.4 Grammar Writing Paradigms

The APN in Figure 1 illustrates several grammar writing paradigms. The situation in which an initial prefix string (*a* or *b*) satisfies a constituent (*P*), but can be followed by optional suffix strings (*cd* or *ce*) occurs frequently in natural language grammars. For example, noun phrase heads in English have optional prenominal and postnominal modifiers. The synchronous disjunction at *P* allows the local role of *a* or *b* to change, while preserving its interpretation as part of a *P*. It is also simple to encode optional prefixes.

Another common situation in natural language grammars is specialization of a constituent based on some internal *feature*. Noun phrases in English, for example, can be specialized by case; verb phrases can be specialized as participial, tensed or infinitive. In Figure 1, node *S* is a specialization which represents "*T*s with *d*-ness or *e*-ness, but not *f-ness*." The specialization is constructed by a synchronous conjunction of features that arise from subtrees somewhere below the node to be specialized.

The APN model also provides for node outputs to be partitioned into independent classes for the purposes of the activation algorithm. The nodes in the classes form *levels* in the network and represent orthogonal systems of classification. The cascading of expectations from different levels can implement context-sensitive behaviors such as feature agreement and semantic selectional restrictions. This is described in Jones (forthcoming). In the next section, we will introduce a grammar writing paradigm to represent movement, another type of non-context-free behavior.

5. It is interesting to speculate on the consequences of various relaxations of this assumption. Fundamental limitations in the allocation of sources may be related to limitations in short term memory (or buffer space in deterministic models; see Marcus, 1980). Linguistic constraints based on constituent length could be related to source decay. Some syntactic garden path behavior might be related to accelerated source decay caused by inhibition from a competing hypothesis. Anything more than a footnote is premature at this time.

3. MOVEMENT

From the APN perspective, movement (limited here to left-extraposition) necessitates the endogenous reactivation of a trace that was created earlier in the process. To capture the trace so that expectations for its reactivation can be posted, we use the following type of rule: (seq ($ *v1* ... *X*...) ($ *v2* ... (and *X X-src Y*) ...). When an instance, *X0*, first activates this rule, *v1* is bound to *X0*; the second occurrence *X* in the rule is constrained to match instances of *X0*, and expectations for *X0*, *X-src* and *Y* are created. No new exogenous source can satisfy the synchronous conjunction; only an endogenous *X-src* can. The rule is similar to the notion of an *X* followed by a *Y* with an *X* hole in it (cf. Gazdar, 1982).

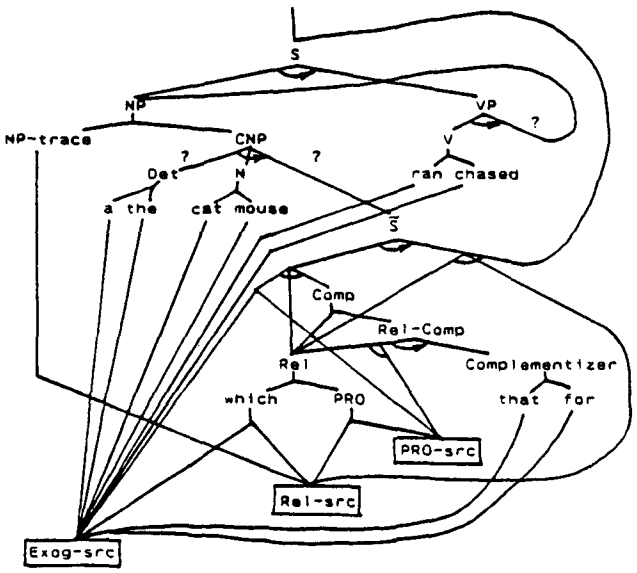


Figure 4. A Grammar for Relative Clauses

Figure 4 defines a grammar with an NP hole in a relative clause; other types of left-extraposition are handled analogously. Our treatment of relatives is adapted from Chomsky and Lasnik (1977). The movement rule for $\bar{S}$ is: (seq ($ *v1* (and *Comp Rel* (or *Exog-src PRO-src*)) ($ *v2* (and *Rel Rel-src S*))). The rule restricts the first instance of *Rel* to arise either from an exogenous relative pronoun such as *which* or from an endogenously generated (phonologically null) pronoun *PRO*. The second variable is satisfied when *Rel-src* simultaneously reactivates a trace of the *Rel* instance and inserts an NP-trace into an *S*.

It is instructive to consider how phonologically null pronouns are inserted before we discuss how movement occurs by trace insertion. The phrase, $[_{NP}$ the mouse $[_{\bar{S}}$ PRO$_i$ that ...]], illustrates how a relative pronoun *PRO* is inserted. Figure 5(a) shows the network after parsing *the cat*. When the complementizer *that* appears next in the input, *PRO-src* receives inhibition (marked by downward arrows in Figure 5(b)) from *Rel-Comp0*. Non-exogenous

sources such as *PRO-src* and *Rel-src* are activated in contexts in which they are expected and then receive inhibition. Figure 5(c) shows the resulting network after *PRO-src* has been activated. The inserted pronoun behaves precisely as an input pronoun with respect to subsequent movement.

The trace generation necessary for movement uses the same insertion mechanism described above. Figures 6(a)-(d) illustrate various stages in parsing the phrase, $[_{NP}$ the cat $[_{\bar{S}}$ which$_i$ $[_S$ t_i ran]]]. In Figure 6(a), after parsing *the cat which*, synchronous expectations are posted for an S which contains a reactivation of the *Rel0* trace by *Rel-src*. The signal sent to S by *Rel-src* will be in the form of an *NP* (through *NP-trace*).

Figure 6(b) shows how the input of *ran* produces inhibition on *Rel-src* from *S1*. The inhibition on *Rel-src* causes it to activate (just as in the null pronoun insertion) to try to satisfy the current contextual expectations. Figure 6(c) shows the network after *Rel-src* has activated to supply the trace. The only remaining problem is that *Rel-src* is actively inhibiting itself through $\bar{S}0$.[6] When *Rel-src* activates again, new instances are created for the inhibited nodes as they are re-activated; the uninhibited nodes are simply rebound. The final structure is shown in Figure 6(d).

It is interesting that the network automatically enforces the restriction that the relative pronoun, complementizer and subject of the embedded sentence cannot all be missing. *PRO* must be generated before its trace can be inserted as the subject. Furthermore, since expectations are strongest for the first link of a sequence, expectations will be much weaker for the *VP* in the relative clause (under S under $\bar{S}$) than for the top-level *VP* under *S0*.

The fact that the *device* blocks certain structures, without explicit well-formedness constraints, is quite significant. Wherever possible, we would like to account for the complexity of the data through the composite behavior of a universal device and a simple, general grammar. We consider the description of a device which embodies the appropriate principles more parsimonious than a list of complex conditions and filters, and, to the extent that its architecture is independently motivated by processing (i.e., performance) considerations, of greater theoretical interest.[7]

As we have seen, certain interpretations can be suppressed by expectations from elsewhere in the network. Furthermore, the occurrence of traces and empty constituents is severely constrained because they must be supplied by endogenous sources, which can only support a single constituent at any given time. For NP movement, these two properties of the device, taken together, effectively enforce Ross's Complex NP Constraint (Ross, 1967), which states that, "No element contained in a sentence dominated by an NP with a lexical head noun may be moved out of that NP by a transformation."

To see why this constraint is enforced, consider the two kinds of sentences that an NP with a lexical head noun might dominate. If the embedded sentence is a relative clause, as in, $[_{NP}$ the rat $[_{\bar{S}}$ which$_i$ $[_S$ the cat $[_{\bar{S}}$ which$_j$ $[_S$ t_j chased t_i]] likes fish]]], then *Rel-src* cannot support both traces. If the embedded sentence is a noun complement (not shown in Figure 4), as in, $[_{NP}$ the rat $[_{\bar{S}}$ which$_i$ $[_S$ he read a report $[_{\bar{S}}$ that $[_S$ the cat chased t_i]]]]], then there is only one trace *in the intended interpretation*, but there is nondeterminism during parsing between the noun complement and the relative clause interpretation. The interference causes the trace to be bound to the innermost relative pronoun in the relative clause interpretation.[8] Thus, the combined properties of the device and grammar consistently block those structures which violate the Complex NP Constraint. Our preliminary findings for other types of movement (e.g., Subject-auxiliary Inversion, *Wh*-movement, and Raising) indicate that they also have natural APN explanations.

4. IMPLEMENTATION and FUTURE DIRECTIONS

Although the research described in this summary is primarily of a theoretic nature, the basic ideas involved in using APNs for recognition and generation are being implemented and tested in Zetalisp on a Symbolics Lisp Machine. We have also hand-simulated data on movement from the literature to design the theory and algorithms presented in this paper. We are currently designing networks for a broad coverage syntactic grammar of English and for additional, cascaded levels for NP role mapping and case frames. The model has also been adapted as a general, context-driven problem solver, although more work remains to be done.

We are considering ways of integrating iterative relaxation techniques with the rule-based framework of APNs. This is particularly necessary in helping the network to identify *expectation coalitions*. In Figure 5(a), for example, there should be virtually no expectations for *Rel-src*, since it cannot satisfy any of the dominating synchronous conjunctions. Some type of non-activating feedback from the sources seems to be necessary.

5. SUMMARY

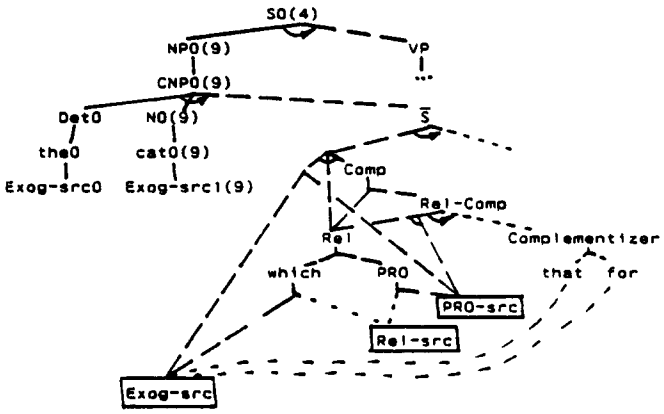
Recent linguistic theories have attempted to induce general principles (e.g., CNPC, Subjacency, and the Structure Preserving Hypothesis) from the detailed structural descriptions of earlier transformational theories (Chomsky, 1981). Our research can be viewed as an attempt to induce the machine that embodies these principles. In this paper, we have described a class of candidate machines, called *active production networks*, and outlined how they handle movement as a natural way in which machine and grammar interact.

The APN framework was initially developed as a plausible cognitive model for language processing, which would have real-time processing behavior, and extensive

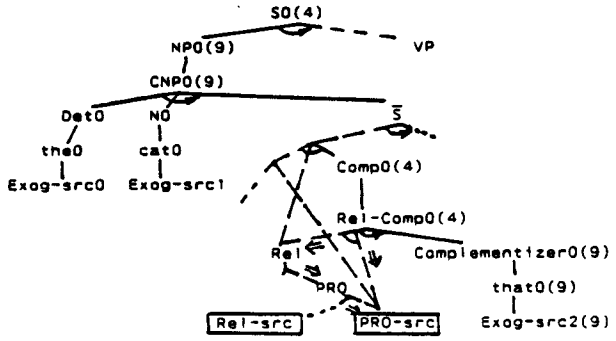
6. Another way of stating this is that the non-synchronicity of the two variables in the pattern has been violated. The self-inhibition of a source occurs in other contexts in the APN framework, even for exogenous sources. In networks that contain left-recursive cycles or ambiguous attachments (e.g., PP attachment), self-inhibition can arise naturally as the result of necessary non-determinism. Re-activation of a self-inhibited source effectively preserves the non-synchronicity of patterns.

7. The work of Marcus (1980) is in this same spirit.

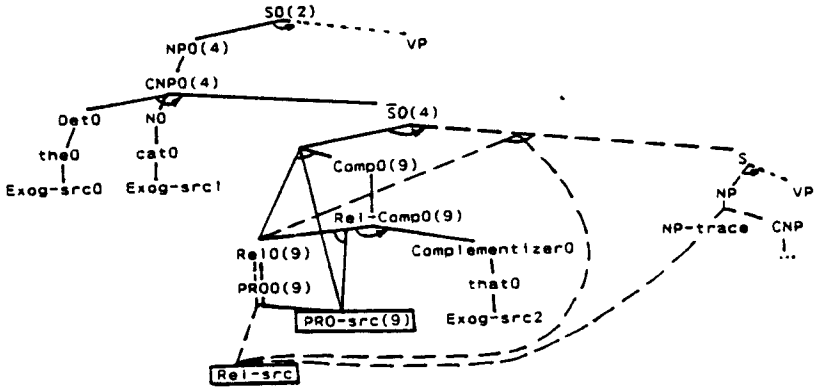
8. Due to recency considerations which relate to expectation strength, traces are bound in a way that preserves nesting.



(a) trace structure after *the cat*



(b) trace structure after *the cat ... that*



(c) trace structure after *the cat* PRO *that*

Figure 5. Relative Pronoun Insertion

contextual processing and learning capabilities based on a formal notion of expectations. That movement also seems naturally expressible in a way that is consistent with current linguistic theories is quite intriguing.

REFERENCES

Anderson, J. R. (1983). *The Architecture of Cognition*, Harvard University Press, Cambridge.

Chomsky, N. (1981). *Lectures on Government and Binding*, Foris Publications, Dordrecht.

Chomsky, N. and Lasnik, H. (1977). "Filters and Control," *Linguistic Inquiry* 8, 425-504.

Fahlman, S. E. (1979). *NETL: A System for Representing and Using Real-World Knowledge*, MIT Press, Cambridge.

Fahlman, S. E., Hinton, G. E. and Sejnowski, T. J. (1983). "Massively Parallel Architectures for AI: NETL, Thistle, and Boltzmann Machines," *AAAI-83 Conference Proceedings*.

Feldman, J. A. and Ballard, D. H. (1982). "Connectionist Models and Their Properties," *Cognitive Science* 6, 205-254.

Gazdar, G. (1982). "Phrase Structure Grammar," *The Nature of Syntactic Representation*, Jacobson and Pullum, eds., Reidel, Boston, 131-186.

Jones, M. A. (1983). "Activation-Based Parsing," *8th IJCAI*, Karlsruhe, W. Germany, 678-682.

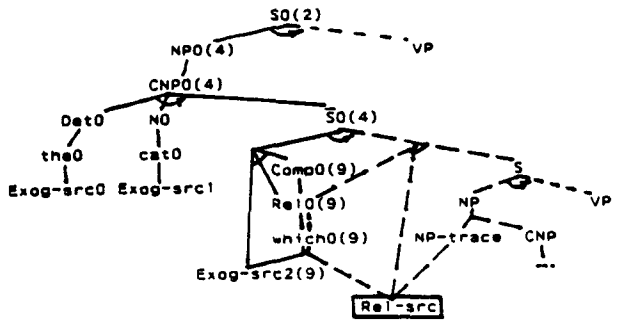
Jones, M. A. (forthcoming). submitted for publication.

Marcus, M. P. (1980). *A Theory of Syntactic Recognition for Natural Language*, MIT Press, Cambridge.

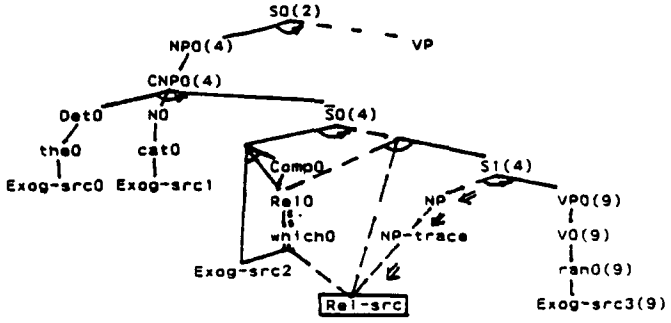
Pereira, F. (1983). "Logic for Natural Language Analysis," technical report 275, SRI International, Menlo Park.

Ross, J. R. (1967). *Constraints on Variables in Syntax*, unpublished Ph.D. thesis, MIT, Cambridge.

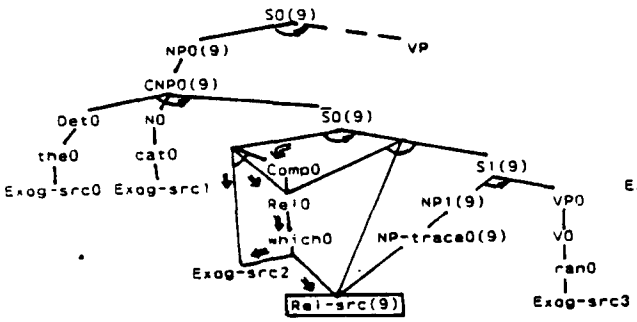
Waltz, D. L. and Pollack, J. B. (1985). "Massively Parallel Parsing: A Strongly Interactive Model of Natural Language Interpretation," *Cognitive Science*, 9, 51-74.



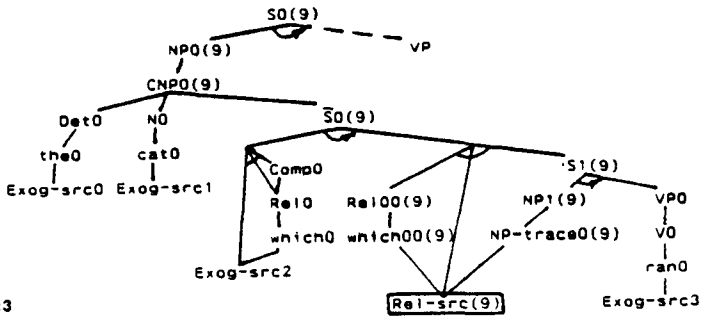
(a) trace structure after *the cat which*



(b) trace structure after *the cat which ... ran*



(c) trace structure just after *the cat which* t *ran*



(d) final trace structure

Figure 6. Parsing Relative Clauses