Proceedings of the Annual Meeting of the Cognitive Science Society

Establishing Long-Distance Dependencies in a Hypbrid Network Model of Human Parsing

Proceedings of the Annual Meeting of the Cognitive Science Society, 15(0)

Stevenson, Suzanne

1993

Establishing Long-Distance Dependencies in a Hybrid Network Model of Human Parsing

Suzanne Stevenson
Department of Computer Science
University of Toronto
Toronto, Ontario M5S 1A4 Canada
suzanne@cs.toronto.edu

Abstract

This paper presents CAPERS, a hybrid spreading activation/marker passing architecture for parsing, whose self-processing network directly represents a parse tree. CAPERS establishes syntactic dependencies through the purely local communication of simple syntactic features within the network. The structural constraints on two nodes in a long-distance syntactic relation are broken down into local components, each of which can be verified entirely between pairs of adjacent nodes along the feature passing path between the two dependent nodes. This method of establishing long-distance syntactic relations, in conjunction with the competitive dynamics of the network, accounts for psycholinguistic experimental data on filler/gap constructions.

1 Introduction

Connectionist approaches to natural language parsing claim to offer a cognitively plausible architecture for modeling human sentence processing. However, while linguistic theory has taken a paradigmatic leap from rule-based systems to constraint-based systems, connectionist parsers have traditionally encoded rule-based linguistic formalisms [COT 89, FAN 85, S&H 85]. This approach is especially surprising given the naturalness of the massively parallel framework for encoding simple features and locally applied constraints. In CAPERS, a hybrid spreading activation/marker passing architecture for parsing, the linguistic knowledge of Government-Binding theory (GB) [CHO 81, RIZ 90] is directly encoded as a set of simultaneous local constraints. Fundamental properties of CAPERS have been shown to explain human behavior in the processing of syntactic ambiguities [STE 93a, 93b, 90]. This paper presents CAPERS' approach to a natural language problem that has received little attention within the connectionist paradigm—the processing of long-distance dependencies.

Long-distance dependencies arise when one position in a sentence depends on another that may be an indefinite distant away. For example, one type of long-distance dependency occurs in filler/gap constructions, in which a WH-phrase (the filler) may be an arbitrary distance from its underlying syntactic position (the gap). Consider the following sentences:

Who did Maya kiss [gap]?

Who did Sara say that Maya kissed [gap]?

The WH-phrase *Who* assumes its role in the sentence based on its underlying position as the object of *kiss* (indicated by [gap]); an unbounded amount of lexical input may intervene between *Who* and this position. Long-distance dependencies pose a problem for rule-based connectionist parsers because of the proliferation of rules needed to encode all of the grammatical structural relations that arise. By contrast, restrictions on dependency relations in CAPERS are captured by a few simple declarative constraints that apply across a range of syntactic constructions.

CAPERS establishes syntactic dependencies through a novel scheme of restricted marker passing, in which grammatical constraints restrict the communication of features between nodes. The success of the approach relies on the insight that the structural constraints on any relation between two nodes in a parse tree can be broken down into local components. The set of local restrictions can then be verified entirely between pairs of directly neighboring nodes along the path in the tree between the two dependent nodes. The restricted feature passing mechanism handles a range of syntactic relations from GB, including movement, agreement, Case marking, and theta role assignment. The integration of simple symbolic features into a connectionist framework, along with appropriate restrictions on the featural communication, allows CAPERS to capture even long-distance syntactic relations in a linguistically and psychologically motivated way.

Because of the amount of relevant psycholinguistic data, the discussion here focuses solely on WH constructions. These examples demonstrate CAPERS' ability to establish long-distance dependencies in a self-processing network parser. Furthermore, CAPERS is shown to account for a range of human behaviors in parsing filler/gap constructions. Rule-based approaches that have modeled some of these behaviors have relied on construction-specific rules to handle the different long-distance structural relations [C&L 91, JON 87, JUR 91]. By contrast, CAPERS accounts for the data with a uniform processing mechanism, whose behavior falls out from independently motivated properties of the model.

The paper is organized as follows. Section 2 briefly presents the CAPERS approach to parsing. Section 3 describes the restricted symbolic feature passing mechanism, and Section 4 explains how this is used to associate a filler and its gap. Section 5 presents CAPERS' account of the psycholinguistic data. Section 6 concludes with a statement of the contribution of CAPERS as a cognitive model of human filler/gap processing.

2 An Overview of CAPERS

Since only a brief description of the model is possible, this section gives a high-level view of parsing in CAPERS, focusing on the properties that distinguish it from traditional connectionist approaches. CAPERS is a hybrid connectionist model that integrates simple symbolic processing with numeric spreading activation, within a self-processing network that directly represents a parse tree. Symbolic features determine the grammaticality of potential attachments, while numeric activation plays the crucial role of weighing the relative strengths of the valid alternatives. CAPERS dynamically creates the network of alternative attachments by allocating processing nodes in response to the input.

When an input token is read, the parser activates two types of nodes: phrasal nodes, or *p-nodes*, to represent the current input phrase, and attachment nodes, or *a-nodes*, to represent the potential attachments of the new phrase. Each a-node connects to exactly two p-nodes, an XP and an X or X′, that are potential *sisters* in the parse tree; compare the network of Figure 1(a) with the tree that it represents in Figure 1(b). Since multiple attachments for a p-node may be grammatical, a competitive output function [REG 87] weighs the evidence for an attachment, and proportionately allocates activation from the p-node to its a-nodes. The competition mechanism necessitates restrictions on the network structure: New a-nodes can only represent an attachment between the current phrase and the right edge of the developing parse tree, as in Figure 1(a).[1]

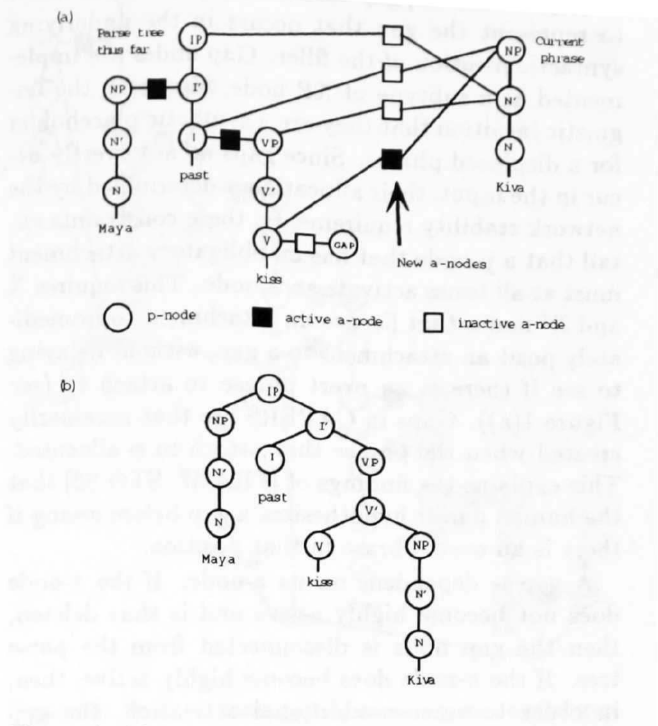

Figure 1: (a) Network for "Maya kissed Kiva," after allocating the $\overline{X}$ phrase for *Kiva*. (b) Parse tree represented by the network in (a).

Once the current phrase is connected to the existing network, each processing node iteratively updates and outputs its symbolic features and numeric activation, until the network settles into a stable state representing the attachment decisions for the current phrase.[2] The self-processing loop ensures that each p-node activates a number of a-nodes consistent with its grammatical properties. For example, a p-node that takes one obligatory complement (such as the V in Figure 1(a)) must activate exactly one a-node, while a p-node that takes an optional specifier may activate zero or one a-node. When the network stabilizes, the set of active a-nodes represents the current parse tree attachments, and the inactive a-nodes are deleted. At this point, the next input token is read and the process is repeated. At the end of a successful parse, the grammatical constraints on each node are satisfied, and the phrases and their attachments form a connected network representing the final parse tree.

---

[1] Maintaining the required network structure entails the use of a parsing stack [STE 93a,b]. The stack is *not* a global control mechanism; it is in fact a single p-node.

[2] CAPERS stabilized in over 98% of nearly 1400 attachment simulations, requiring 10–70 iterations in each case.

To parse a filler/gap construction, CAPERS needs to represent the gap that occurs in the underlying syntactic position of the filler. Gap nodes are implemented as a subtype of XP node, mirroring the linguistic intuition that they are a syntactic placeholder for a displaced phrase. Since gaps do not overtly occur in the input, their allocation is determined by the network stability requirements; these constraints entail that a p-node that has an obligatory attachment must at all times activate an a-node. This requires X and X′ nodes that license an attachment to immediately posit an attachment to a gap, without delaying to see if there is an overt phrase to attach to (see Figure 1(a)). Gaps in CAPERS are thus necessarily created when the phrase they attach to is allocated. This explains the findings of [FRA 87, STO 86] that the human parser hypothesizes a gap before seeing if there is an overt phrase in that position.

A gap is dependent on its a-node. If the a-node does not become highly active and is thus deleted, then the gap node is disconnected from the parse tree. If the a-node does become highly active, then, in order to receive additional activation, the gap must be associated with a filler through CAPERS' symbolic feature passing process.

3 Symbolic Feature Passing

Symbolic features are implemented as attributes with atomic values; any GB constraints on the features of a node are encoded as simple equality tests on the values of its attributes. The challenge for CAPERS is verifying the structural configuration that must hold between two nodes in a given syntactic relation—for example, for subject-verb agreement or for associating a filler with a gap. A self-processing network has no global perspective on its own structure; even non-local parsing decisions must be made solely on the basis of local communication. Furthermore, CAPERS' limited symbolic capabilities are unable to build in global information by creating feature structures that could encode the history of a feature passing path.

A solution for achieving local verification of structural constraints in CAPERS exploits two facts: (1) A syntactic relation between two nodes involves features that must be assigned or shared between them; and (2) Features passed between nodes must travel through the network, which is a direct representation of the parse tree structure. Thus, CAPERS can enforce even non-local structural constraints on a syntactic relation by constraining each segment of a feature passing path according to the grammatical restrictions that apply to the particular feature being communicated. The feature passing restrictions of CAPERS were derived directly from the linguistic theory, by analyzing GB's structural constraints into locally applicable primitives [STE 93a]. Symbolic features are defined within a grammatical hierarchy, in which the features inherit the appropriate local communication constraints. Each node in the parsing network calls the same feature passing routine, which applies a feature's communication restrictions to determine whether the node can pass that feature to each of its neighbors. This uniform feature passing mechanism ensures both local and non-local structural constraints on syntactic relations.

GB constraints on filler/gap constructions state that a filler and its gap must be *coindexed* under a certain structural configuration. Coindexation is a grammatical process by which two linguistic entities are associated to indicate that they refer to the same object in the world. The structural constraint on coindexing a gap with a filler has three components. First, the filler must occur in a particular tree position—attached to the left of an X′ node. Second, there can be no *barriers*—nodes with particular features—between the filler and the gap. The third restriction is the general condition of Minimality [RIZ 90], which states that a node X can only have a certain relation to another node Y if there is no closer node Z that can have that relation to Y. That is, the filler must be the node at a minimal distance from the gap that satisfies the first two syntactic constraints on the coindexation relation.

Each of these conditions maps directly to a simple feature passing constraint within CAPERS. First, the tree structure position of the filler is a simple check on the features of a potential destination node. Second, a node with certain specified features acts as a barrier by not being able to pass features beyond itself. Finally, Minimality is implemented by not letting a node to which a feature can validly apply pass that feature on to its neighbors. Thus, the feature passing mechanism of CAPERS is a simple and elegant computational model of GB's declarative constraints on filler/gap coindexation.[3]

4 The Coindexing Mechanism

Analogous to the use of a-nodes to explicitly represent parse tree attachments, CAPERS uses coindexing nodes, or c-nodes, to overtly represent coindexation relations. Each c-node connects to two p-nodes,

---

[3] These conditions represent the *antecedent government* constraint on gap constructions [RIZ 90]; the other constraint proposed by Rizzi, *head government*, is implemented in CAPERS with analogous feature passing constraints.

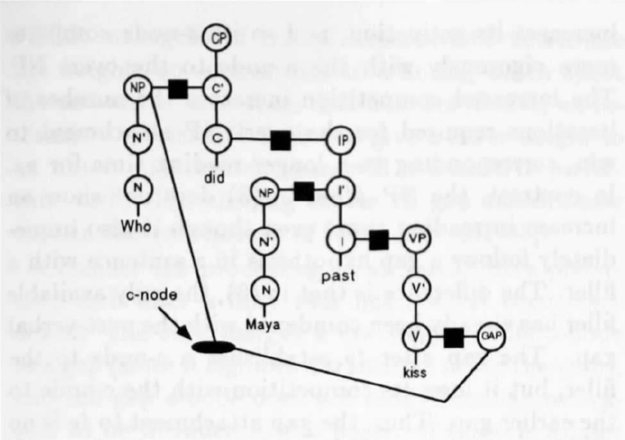

Figure 2: When *kiss* is allocated, a gap is automatically created in object position. The gap sends features to *Who*, triggering creation of the c-node.

a gap and its filler. Since multiple c-nodes may be established for a given p-node, a c-node must compete for numeric activation from its two p-nodes. The similarity of c-nodes to a-nodes allows them to be implemented as a subtype of a-node.

C-nodes are allocated when coindexing features are successfully passed through the tree from a gap to a potential filler. Consider the sentence "Who did Maya kiss." After CAPERS has attached the VP for *kiss*, the network has the structure shown in Figure 2. The gap in object position (the sister to the V) must be coindexed with a filler; it thus sends a feature through the tree that will trigger the establishment of a c-node. Since each segment of the feature passing path between the gap and *Who* complies with the grammatical restrictions described above, the feature will arrive successfully at *Who*.[4] When *Who* receives the feature from the gap, it allocates a c-node, sends the c-node's address to the gap, sets up its own link to the c-node, and begins to activate it competitively. When the gap receives the c-node's address, it too sets up a link to the c-node and begins to activate it. Since there are no competing c-nodes, the c-node becomes highly active, representing a coindexation of the filler and gap.

The same mechanism that establishes this within-clause coindexation relation handles longer distance filler/gap pairs as well. Consider the sentence "Who did Sara say that Maya kissed," in which the filler and gap occur in different clauses. Recall that Minimality prevents coindexing features from continuing past a valid recipient node. Figure 3 shows how the Minimality constraint forces CAPERS to use gap g_1 (which, like all other gaps, is automatically postulated) to achieve a long-distance coindexation relation between the filler and gap g_2.

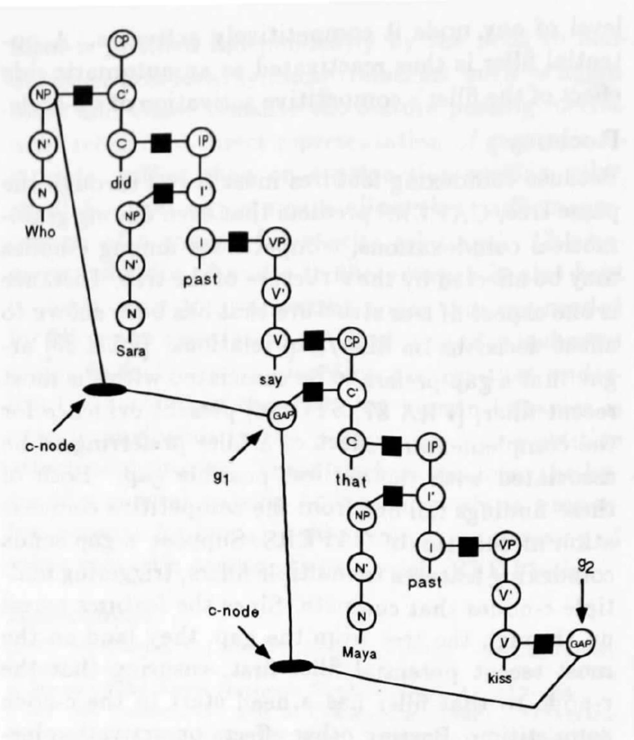

Figure 3: Features from gap g_2 cannot pass beyond gap g_1, due to Minimality. But since g_1 has sent coindexing features to *Who*, *Who*–g_1–g_2 form a coindexing chain linking the filler to its underlying position.

5 Psycholinguistic Accounts

Grammaticality Effects

The grammatical nature of the coindexation mechanism conforms with psycholinguistic data. For example, [STO 86] found that syntactic constraints are used to immediately rule out a gap in an ungrammatical position. Although CAPERS posits gaps automatically in *all* positions, regardless of their structural relation to a filler, it is unable to establish an ungrammatical coindexation relation. The lack of coindexing will thus rule out a gap that is in an ungrammatical relation to the available filler. CAPERS also explains the results of experiments in which all and only grammatical fillers are reactivated by semantic priming at the position of a gap [NIC 88]. In CAPERS, all and only grammatical fillers receive features from a gap, and each such filler establishes a c-node that competes for activation. The competitive mechanism of CAPERS requires that a p-node's input function necessarily relies on the activation

---

[4] The features do not violate Minimality by passing beyond *Maya*; even though it is a sister to an X', it is in an *argument* position. Only a *non-argument* node (e.g., the sister to a C') can receive these coindexing features.

level of any node it competitively activates. A potential filler is thus reactivated as an automatic side effect of the filler's competitive activation of a c-node.

Recency

Because coindexing features must travel through the parse tree, CAPERS predicts that even among grammatical coindexations, competitions among c-nodes may be affected by the structure of the tree. Distance is one aspect of tree structure that has been shown to affect decisions on filler/gap relations. [FCR 83] argue that a gap prefers to be associated with the most recent filler; [FRA 87, STO 86] present evidence for the complementary effect of a filler preferring to be associated with the earliest possible gap. Both of these findings fall out from the competitive coindexation mechanism of CAPERS. Suppose a gap sends coindexing features to multiple fillers, triggering multiple c-nodes that compete. Since the features travel up through the tree from the gap, they land on the most recent potential filler first, ensuring that the c-node to that filler has a head start in the c-node competition. Barring other effects on activation levels, the c-node from the gap to the most recent filler will thus win the competition. Similarly, if multiple gaps send features to the same filler, the c-node from the closest gap to the filler will win its competition.

Filled Gap Effect

The conclusion in [STO 86] that the human parser postulates gaps as early as possible is based on evidence of increased reading times for an overt object in the place of a potential gap (the filled gap effect). She compared sentences such as the following:

My brother wanted to know...
(1) if Ruth will bring us home to Mom at lunch.
(2) who Ruth will bring us home to [gap] at lunch.
(3) who Ruth will bring [gap] home to Mom at lunch.

The reading time for *us* is significantly longer in (2) than in (1), indicating that the parser is having to revise an initial hypothesis of a gap at the post-verbal location. However, in comparing (1) to (3), there is no corresponding slow-down at the word *Mom*. Evidently, the fact that in (3) the filler is coindexed with the post-verbal gap prevents a gap hypothesis after *to* from interfering with the overt NP *Mom*.

CAPERS explains each of these results. In all three sentences, a post-verbal gap is allocated along with the V node for *bring*, but only in (2) or (3) does this gap establish a coindexing relation with a filler. When the overt NP *us* occurs next, as in (1) or (2), its a-node to the verb competes with the gap attachment. In sentence (1), the gap is very weak, and the a-node to the overt NP wins immediately. However, in sentence (2), the coindexation of the gap greatly increases its activation, and so its a-node competes more vigorously with the a-node to the overt NP. The increased competition increases the number of iterations required for the overt NP attachment to win, corresponding to a longer reading time for *us*. In contrast, the NP *Mom* in (3) does not show an increase in reading time, even though it also immediately follows a gap hypothesis in a sentence with a filler. The difference is that in (3), the only available filler has already been coindexed with the post-verbal gap. The gap after *to* establishes a c-node to the filler, but it loses its competition with the c-node to the earlier gap. Thus, the gap attachment to *to* is no stronger in (3) than in (1), and there is no increase in time needed to attach the NP *Mom*.

Somewhat surprisingly, [STO 86] found that there is no increase in reading time for an overt subject after a filler; the reading time for *Ruth* is the same in sentences (1) through (3). To explain both the increased reading time for an overt object and the lack of an increase for an overt subject, [STO 86] concludes that subject and object positions following a WH-phrase must be handled by distinct parsing routines. By contrast, in CAPERS, the differences in the processing of gaps in subject and object position follow directly from the use of a uniform mechanism on different positions in the tree. In GB, the subject of a sentence attaches to the left of the sentential phrase (Infl). Thus in CAPERS, a subject gap is allocated when the Infl phrase is allocated, which occurs at the tense features of the verb. If there is an overt subject, the Infl phrase and its gap hypothesis are allocated *after* it. The overt subject will win the a-node competition with the gap before the gap has the chance to establish a coindexing relation with the filler. CAPERS thus predicts no difference in the processing of *Ruth* in any of the above sentences.

Verb Frame Preferences

CAPERS' competitive attachment and coindexation mechanisms also account for the affect of verb preferences on the ease of processing various filler-gap constructions. [TSC 85] compared sentences with all combinations of transitive and intransitive preference verbs, and early (immediately post-verbal) and late gaps. In sentences with transitive preference verbs, late gaps are harder for the human parser than early gaps; in sentences with intransitive preference verbs, early gaps are harder than late gaps. [C&T 88] observe that the difference in difficulty for the early and late gap intransitive cases is greater than that for the transitive cases. In their words, "the penalty for missing an early gap [the intransitive cases] was larger than the penalty for having to reassign a previously assigned filler [the transitive cases]." In CA-

PERS, strengths of lexical expectations determine the weights on connections to a-nodes, which affect the amount of activation that a-nodes receive; an intransitive preference verb will give a lower weight to an object attachment than will a transitive preference verb. The ensuing effects on gap attachments explain the conclusions of [TSC 85, C&T 88].

Consider the processing of an intransitive preference verb after a filler that has not yet been coindexed. The verb assigns a low weight to its a-node to a gap (since it signifies a transitive use of the verb), and the gap a-node does not receive enough activation to be included in the parse. If there is a later gap in the sentence that can be coindexed with the filler, then parsing proceeds normally. However, if the sentence has no other grammatical gap possibility (as in the early gap cases), then the sentence ends with the filler violating its constraints, since it does not activate a c-node. Recovery from missing the early gap is not straightforward; in fact, it requires adding a special strategy to CAPERS to find a legal position in which to re-allocate a gap under these circumstances. Thus, with these verbs, an early gap sentence is much harder than a late gap sentence.

In contrast, a transitive preference verb assigns a high weight to its (optional) gap attachment, ensuring that the a-node wins and the gap establishes a c-node to the filler. An early gap sentence in which this coindexation relation is correct is easily parsed. However, a late gap sentence causes difficulty. A later obligatory gap will send features to the single available filler, which establishes a second c-node to the new gap. The second c-node, although competing against an earlier, established one, is stronger because the later gap is obligatory while the earlier one is optional. The new c-node thus wins the competition, changing the coindexation of the single available filler from the earlier, optional gap to the later, obligatory gap. The c-node competition and revised coindexation make these late gap sentences harder than the early gap ones. However, the increase in difficulty is less than that arising in the intransitive preference cases, since here the normal competitive mechanisms accomplish the necessary revision.

6 Conclusions

Since psycholinguistically observed behavior follows from independently motivated properties of the model, CAPERS provides an explanatory theory of human filler/gap processing. The grammatically restricted feature passing mechanism successfully localizes the establishment of long-distance dependencies within a self-processing network. The mechanism is justified independently by the need to handle other non-local syntactic relations, such as agreement and Case. Because the feature passing mechanism relies on a direct representation of general constraints, rather than on construction-specific rules, small featural adjustments allow the uniform processing of a range of syntactic relations. The numeric behavior of nodes in the network is also kept uniform, since the coindexing nodes that are needed for filler/gap association are a subtype of attachment node. In fact, the coindexation process that underlies the account of the observed human behavior is exactly analogous to the parser's basic competitive attachment process. These mechanisms form the basis of a general parsing architecture whose properties account for human behavior in the processing of other syntactic constructions as well [STE 93a,b].

References

[C&L 91] Cardie, C., and W. Lehnert (1991). "A Cognitively Plausible Approach to Understanding Complex Syntax." *Proceedings of the Ninth National Conference on AI*, 117–124.

[C&T 88] Carlson, G., and M. Tanenhaus (1988). "Thematic Roles and Language Comprehension." *Syntax and Semantics* **21**, 263–288.

[CHO 81] Chomsky, N. (1981). *Lectures on Government and Binding: The Pisa Lectures*. Dordrecht: Foris.

[COT 89] Cottrell, G. W. (1989). *A Connectionist Approach to Word Sense Disambiguation*. Los Altos, CA: Morgan Kaufmann.

[FAN 85] Fanty, M. (1985). "Context-free parsing in connectionist networks." Technical Report TR174, University of Rochester.

[FRA 87] Frazier, L. (1987). "Sentence Processing: Evidence from Dutch." *Natural Language and Linguistic Theory* **5**, 519–559.

[FCR 83] Frazier, L., C. Clifton, and J. Randall (1983). "Filling Gaps: Decision principles and structure in sentence comprehension." *Cognition* **13**, 187–222.

[JON 87] Jones, M. (1987). "Feedback as a Coindexing Mechanism in Connectionist Architectures." *Proceedings of the Tenth International Joint Conferences on AI*, 602–610.

[JUR 91] Jurafsky, D. (1991). "An On-Line Model of Human Sentence Interpretation." *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*, 449–454.

[NIC 88] Nicol, J. (1988). "Coreference processing during sentence comprehension: Reactivating antecedents of empty categories during sentence processing." Doctoral dissertation, MIT.

[REG 87] Reggia, J. (1987). "Properties of a Competition-Based Activation Mechanism in Neuromimetic Network Models." *Proceedings of the First International Conference on Neural Networks*, II-131–II-138.

[RIZ 90] Rizzi, L. (1990). *Relativized Minimality*. MIT Press.

[S&H 85] Selman, G., and G. Hirst (1985). "A Rule-Based Connectionist Parsing Scheme." *Proceedings of the Seventh Annual Conference of the Cognitive Science Society*, 212–219.

[STE 93a] Stevenson, S. (1993a). "A Constrained Active Attachment Model for Resolving Syntactic Ambiguities in Natural Language Parsing." Doctoral dissertation, Computer Science Department, University of Maryland, College Park.

[STE 93b] Stevenson, S. (1993b). "A Competition-Based Explanation of Syntactic Attachment Preferences and Garden Path Phenomena." To appear in *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*.

[STE 90] Stevenson, S. (1990). "A Parallel Constraint Satisfaction and Spreading Activation Model for Resolving Syntactic Ambiguity." *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*, 396–403.

[STO 86] Stowe, L. (1986). "Parsing WH-constructions: evidence for on-line gap location." *Language and Cognitive Processes* **1:3**, 227–245.

[TSC 85] Tanenhaus, M., L. Stowe, and G. Carlson (1985). "The interaction of lexical expectation and pragmatics in parsing filler-gap constructions." *Program of the Seventh Annual Conference of the Cognitive Science Society*, 361–365.