Incremental processing and infinite local ambiguity

Vincenzo Lombardo
Dipartimento di Informatica
Università di Torino
Corso Svizzera 185
10149 Torino
Italy
vincenzo@di.unito.it

Patrick Sturt
Centre for Cognitive Science
University of Edinburgh
2 Buccleuch Place
Edinburgh EH8 9LW
Scotland
sturt@cogsci.ed.ac.uk

Abstract

In incremental parsing, infinite local ambiguity occurs when the input word can be combined with the syntactic structure built so far in a infinite number of ways. A common example is left recursion (e.g. "railway station clock" or "his sister's boyfriend's shirt"), where local information cannot tell us the depth of embedding of the left descendent chain of nodes. From the processing point of view, infinite local ambiguity causes a technical problem, which a model must solve in order to implement incrementality fully. This paper provides a general solution to the problem of infinite local ambiguity, by introducing the concept of Minimal Recursive Structure. We give two examples of parsers in which the solution is used.

Introduction

Strong Incrementality

Many theories of human sentence processing assume that language comprehension is constrained by what we will call strong incrementality. According to this constraint, the processor reads its input strictly from left to right, maintaining a fully connected structure at each state (cf. the Left-to-Right constraint of Frazier and Rayner (1988), or the Incremental Licensing principle of Gorrell (1995)). Constraining processing by strong incrementality gives at least two advantages to the human language processing system. First, assuming a semantics which can be applied to the kind of partial structures built by an incremental parser, it allows for the interpretation of incomplete sentence fragments, which are common in everyday speech. Secondly, the constraint avoids the computational cost of maintaining large numbers of unstructured items in working memory. There is a great deal of psychological evidence in favour of the incremental structuring of input. Experiments conducted by Marslen-Wilson (1975) showed that humans could shadow and interpret speech at a delay of only 250ms. Furthermore, despite proposals that structure building is head-driven, a strategy which is not strongly incremental, (as is proposed, for example, by Abney (1989) and Pritchett (1992)), there is an increasing body of experimental evidence, particularly concerning the processing of head-final constructions, which suggests that the processor eagerly assembles and attaches constituents before the head is processed (see, for example, Yamashita (1994), Bader and Lasser (1994), and Hemforth, Konieczny, and Scheepers (1994)). Further evidence comes from the human ability to identify a referent in a spatial context as soon as some distinctive attribute appears in the phrase that describes it, and this can occur well before the appearance of the actual head (Eberhard et al. 1995).

Parsing models and strong incrementality

The necessity of keeping the syntactic structure fully connected strongly constrains the form of the parsing model. There are several algorithms, originating from computer science practice, which have been imported into psycholinguistics to provide the processing mechanisms for human parsing models.

Abney and Johnson (1991) evaluate the psycholinguistic relevance of three common parsing schemas for context free grammar: top-down, bottom-up and left-corner. The evaluation is in terms of the requirements of working memory space and the generation of local ambiguities. Roughly speaking, top-down parsing predicts the lower levels of the syntactic tree by expanding the structure from the upper levels (starting from the root), and scans actual words by matching them on preterminal symbols. The predictive component, if not immediately checked against the input data, can cause the exploration of useless analyses. Also, it suffers from the problem of left-embedding structures (such as NP $\rightarrow$ NP's N), which theoretically cause non terminating computations, or unbounded space requirements. Bottom-up parsing proceeds in the opposite direction, by assembling constituents in larger and larger units, starting from words: it is entirely data-driven and does not embody any predictive component. Bottom-up parsing requires a large (theoretically unbounded) space when dealing with right-embedding constructions (like NP $\rightarrow$ art n PP, PP $\rightarrow$ p NP), as it can assemble a connected syntactic structure only at the very end of the analysis. On the other hand, it parses left-embedding constructions easily. Finally, left-corner parsing is a bottom-up strategy which projects upper levels of the tree as soon as the leftmost daughter has been parsed. Because of this, it outperforms bottom-up parsing, because the processor can build at least a partially connected structure before the end of the constituent, thus reducing the number of unstructured items.

Abney and Johnson evaluate psycholinguistic relevance with the capability of the parsing algorithm to reflect human performances: the larger the memory space requirements and

the local ambiguities,[1] the more difficult parsing becomes. According to their analysis, the left-corner strategy exhibits the best performance profile, as it can explain the human difficulty in parsing center-embedding constructions (requiring a large memory space) in comparison with left and right-embedding constructions. In particular, top-down parsing has large space requirements for left and center-embeddings, and bottom-up parsing has large space requirements for center and right-embedding, thus failing to reflect the differences indicated by psycholinguistic evidence.

Of the three strategies, only top-down parsing guarantees the permanent connectedness of syntactic structure. Hence, if we reduce the unbounded space requirements of top-down parsing with left-embedding constructions, we have a parser which is strongly incremental and psycholinguistically valid.

An alternative solution which has been pursued in the literature to yield strong incrementality is to use a syntactic formalism other than context free grammar. For example, strong incrementality holds for some approaches to categorial grammar, such as that of Milward (1995).

In this paper we describe a general method for reducing the space requirements and the perception of local ambiguities in top-down parsing, the strategy that naturally realizes strong incrementality. The method strongly relies on a lexicalized version of context free grammar. Its generality consists in its applicability to a series of parsing models. We will firstly describe the method in terms of a general parsing algorithm, and subsequently see its application in a specific psycholingusitic model.

The problem of infinite local ambiguity

The task of a strongly incremental parser is to make a left-to-right pass of the input, and find a way of connecting each word into the parse tree as it is encountered. We call the chain of nodes which need to be instantiated in order to connect the input word to the structure built so far the "connection path". The problem of infinite local ambiguity is that there is no way of knowing in advance how long the connection path must be. Imagine that the processor has received the sentence fragment "John hates", and has processed it as a subject followed by an incomplete verb phrase.

[$_S$ [$_{NP}$ John] [$_{VP}$ hates .NP]]

This sequence of brackets and symbols represents the partial parse tree. The dot indicates the current position of analysis. In the example, the parser expects the recognition of a NP in the rest of the sentence. Suppose the next input word is "his". There is an infinite number of ways in which this word can be combined with the structure built so far, as the following continuations show:

(1) a. John hates his sister.

b. John hates his sister's boyfriend.

c. John hates his sister's boyfriend's shirt.

d. *etc...*

Local information at "his" cannot determine the length of the connection path (made of NP nodes) from "hates" (or rather, its projection S) to "his"; that is, we do not know the level of embedding of the left recursion.

Left-embedding structures represent a typical case of infinite local ambiguity, because of the left-to-right character of incremental parsers; right-embedding structures can be easily parsed incrementally without introducing infinite ambiguity. Now, as there is empirical evidence that left-embedding structures are fairly easy for humans (for example, Japanese speakers have little trouble interpreting the left recursive structures common in their language (Mazuka et al, 1989)), and are certainly easier than multiple centre embedding constructions, a psycholinguistically valid parsing model must account for this difference.

In the next section we describe a solution to the problem of parsing left-embedding structures, by introducing the notion of Minimal Recursive Structure. The MRS is a left-descendent chain of nodes in which there are no two nodes of the same category. Intuitively, it represents the minimal unit of left recursion. To connect the current input word to the syntactic structure, the parser builds a MRS (which only needs a finite time); if the subsequent input requires a further extension of the left recursive structure, it adjoins further MRS's.

The notion of "Minimal Recursive Structure"

The solution that we propose to the problem exemplified by (1) relies on the notion of minimum effort, and is related to the well-known Minimal Attachment principle (Frazier, 1978). The intuition behind this solution is already present in the literature in some forms. All authors who propose strongly incremental parsers must provide some solution to the problem of infinite local ambiguity, and, as far as we are aware, all solutions proposed rely on the notion of minimal effort, either implicitly or explicitly, in a way which resembles the proposal we describe here. Our contribution is to define this notion, in terms of MRS, and show how it can be implemented in working models.

The basic idea is the following: when the grammar permits an unbounded left descendent chain of nodes, the resulting structure does not have a random pattern, but rather can be seen as a repetition of some specific pattern which we call Minimal Recursive Structure (MRS). The whole chain is formed by a number of MRS's linked to each other.

To define MRS's formally, we first introduce the more general notion of Dotted Partial Structure (DPS). DPS's represent an alternative view of grammar rules as partially instantiated syntactic structures. In the remainder of this section, we will show how DPS's can be used in the implementation of a top-down parser which is able to process structures which exhibit infinite local ambiguity. A similar idea was proposed

[1] As Abney and Johnson point out, given a single grammar, local ambiguity varies according to the parsing strategy used.

by Thompson et al. (1991), who implemented a strongly incremental parser with the rather different goal of improving the efficiency of parallel context-free parsing, also employing a notion of MRS similar to the one we propose here.

A DPS is a syntactic structure where:

a. the leftmost deepest symbol is a lexical item;
b. each branching is licensed by a grammar rule;
c. a dot precedes or follows some symbol of the structure.

For example, given the grammar:

S → NP VP
NP → propn
VP → v NP

we have the following DPS's:

[$_S$ [$_{NP}$ .propn] VP]
[$_{VP}$ .v NP]
[$_{NP}$ .propn]

The dot in the initial position states that no part of the structure has been analyzed yet. The construction of the DPS's of a given grammar is a straightforward process. From the start symbol rules, we expand the leftmost symbol of the right hand side iteratively, until we reach a preterminal category. This occurs for every possible path through the rules. Of course, there is a problem with left recursion, which we deal with below.

Once the grammar is written in DPS form, we can write an algorithm that keeps the structure permanently connected during sentence parsing. For the sake of simplicity, the description of the parser (which follows Earley's style (Earley, 1970)) does not incorporate preferences for local ambiguity. Instead, we assume that each step is performed nondeterministically, and is the correct move. Here is the algorithm:

INITIALIZATION: the parser takes a DPS of the form [$_S$ ...].

Then, for each input word:

PREDICTION: If the dot precedes a non terminal symbol X we replace it with a DPS of the form [$_X$ ...].

SCANNING: If the dot precedes a preterminal symbol x and the input word is of category x, then we advance the dot.

COMPLETION: If the dot precedes a closed bracket, then advance the dot after it.

Finally,

TERMINATION: If we have a DPS of the form [$_S$ ...] and the dot is in the rightmost position, then exit with ACCEPT; otherwise REJECT.

Let us see how it works through an example, "John hates Mary"

INITIALIZATION
[$_S$ [$_{NP}$ .propn] VP]
SCANNING
[$_S$ [$_{NP}$ John.] VP]
COMPLETION
[$_S$ [$_{NP}$ John] .VP]
PREDICTION
[$_S$ [$_{NP}$ John] [$_{VP}$ .v NP]]
SCANNING
[$_S$ [$_{NP}$ John] [$_{VP}$ hates .NP]]
PREDICTION
[$_S$ [$_{NP}$ John] [$_{VP}$ hates [$_{NP}$ .propn]]]
SCANNING
[$_S$ [$_{NP}$ John] [$_{VP}$ hates [$_{NP}$ Mary.]]]
COMPLETION (x 3)
[$_S$ [$_{NP}$ John] [$_{VP}$ hates [$_{NP}$ Mary]]].

In the case of a left recursive grammar, the top down expansion of leftmost symbols never terminates. Thus, the DPS computation stops as soon as some symbol is repeated: the DPS produced up to that moment represents a MRS and is labelled with a unique identifier. These labels are then used to mark the DPS's with the same top symbol. This means that they can be expanded to larger structures. Formally, a MRS is a labelled DPS.

Let us consider the following grammar, which contains both direct and indirect left recursion:

S → NP VP
NP → D n
NP → NP PP
NP → propn
VP → v NP
D → art
D → NP 's
PP → p NP

The DPS's produced are the following:

[$_S$ [$_{NP}$[$_D$ .art] n]1,2 VP]
[$_S$ [$_{NP}$ .propn]1,2 VP]
[$_{NP}$ [$_D$ .art] n]1,2
[$_{NP}$ .propn]1,2
[$_D$ .art]
[$_{VP}$ .v NP]
[$_{PP}$ .p NP]

1: [$_{NP}$ [$_D$ .NP 's] n]1,2
2: [$_{NP}$ .NP PP]1,2

Intuitively, the parser implements the following idea: when it analyzes an input word and encounters a left recursive construction, it builds a MRS, that is a structure with only one

level of recursion, to insert the word. Then, the parser leaves a marker (label) to remember that this minimal structure can be expanded with further MRS's.

The previous algorithm is accordingly augmented. In particular, the completion phase also non-deterministically executes the following code:

If the dot precedes a closed bracket which ends a DPS X and X is marked with L, then the DPS L replaces X in the larger DPS and X replaces the non terminal that follows the dot in L; finally, the dot advances after X.

For example, take the following DPS, where non terminals have subscripts to allow subsequent identification,

[$_{NP1}$ [$_{D1}$ the] child.]1

After completion it becomes

[$_{NP2}$ [$_{D2}$ [$_{NP1}$ [$_{D1}$ the] child] .'s] n]1,2

Let us see how the parser works on the example "Mary hates the child's dog", starting from the input word "the".

[$_S$ [$_{NP}$ Mary] [$_{VP}$ hates .NP]]

PREDICTION

[$_S$ [$_{NP}$ Mary] [$_{VP}$ hates [$_{NP}$ [$_D$ .art] n]1,2]]

SCANNING

[$_S$ [$_{NP}$ Mary] [$_{VP}$ hates [$_{NP}$ [$_D$ the.] n]1,2]]

COMPLETION

[$_S$ [$_{NP}$ Mary] [$_{VP}$ hates [$_{NP}$ [$_D$ the] .n]1,2]]

SCANNING

[$_S$ [$_{NP}$ Mary] [$_{VP}$ hates [$_{NP}$ [$_D$ the] child.]1,2]]

COMPLETION

[$_S$ [$_{NP}$ Mary]
[$_{VP}$ hates [$_{NP}$ [$_D$ [$_{NP}$[$_D$ the] child] .'s] n]1,2]]

SCANNING

[$_S$ [$_{NP}$ Mary]
[$_{VP}$ hates [$_{NP}$ [$_D$ [$_{NP}$[$_D$ the] child] 's.] n]1,2]]

COMPLETION

[$_S$ [$_{NP}$ Mary]
[$_{VP}$ hates [$_{NP}$ [$_D$ [$_{NP}$[$_D$ the] child] 's] .n]1,2]]

SCANNING

[$_S$ [$_{NP}$ Mary]
[$_{VP}$ hates [$_{NP}$ [$_D$ [$_{NP}$[$_D$ the] child] 's] dog.]1,2]]

COMPLETION (x 3)

[$_S$ [$_{NP}$ Mary]
[$_{VP}$ hates [$_{NP}$ [$_D$ [$_{NP}$[$_D$ the] child] 's] dog]]].

The parser implements strong incrementality, as it keeps a fully connected syntactic structure. The parser (and the example) executes non-deterministic steps, and goes straight to the correct solution without taking care of local ambiguity management. The DPS structures contribute to the reduction of local ambiguity due to the use of lexicalization, which introduces a data-driven component into the basic top-down character of the algorithm. Additionally, we no longer have unbounded space requirements in the case of left recursion, since MRS's are instantiated as needed.

To use this basic schema in a parsing model we need to account for local ambiguity management. Theories devised by psycholinguists vary according to the extent to which the human language processing system maintains multiple syntactic analyses in parallel. For example, Frazier (1978), argues that a single analysis is computed, and revised if necessary. On the other hand, parallel models such as that of Gibson (1991) and competitive activation models such as that sketched in MacDonald, Pearlmutter, and Seidenberg (1994) allow the processor to construct and maintain (or activate, respectively) a number of alternative analyses in parallel. A serial architecture maintains only one analysis, which, assuming strong incrementality, must be connected, at any one state. In a parallel architecture, strong incrementality could be interpreted in a number of ways; for example, it could be interpreted as a requirement that all analyses under consideration at any one state should be connected. On the other hand, a weaker interpretation could be that, of all the analyses under consideration at any particular state, at least one should be fully connected.

The non-deterministic algorithm described above can be easily extended to parallel models: it is sufficient to assume that the single steps are executed on each possible analysis by assuming a duplication of structures where local ambiguity occurs. Lombardo (1995, 1996) describes a limited parallel parsing model which also implements the sharing of common structures.

In the next section we present a serial psycholinguistic model, that implements an on-line notion of MRS, without putting the grammar into DPS form.

A psycholinguistic model of parsing infinite local ambiguity

In the previous section, we described how the notion of minimal recursive structure can be used to add a lexical data-driven component to constrain a top-down parser, providing a solution to the problem of infinite local ambiguity. This was achieved by pre-compiling the grammar to produce the full set of DPS structures necessary to connect an arbitrary word with an arbitrary left context. We believe that pre-compilation is useful for the construction of efficient algorithms, and that the derived structures give an intuitively clear idea of the notion of minimal recursive structure. However, pre-compilation is not a necessary condition for dealing with infinite local ambiguity in incremental parsing. This is important, because many psycholinguists subscribe to the *Type Transparency Hypothesis* of Berwick and Weinberg (1985), which states that "grammatical representations are embedded directly into parsers" (see, for example, Pritchett (1992), Gorrell (1995), for arguments in favour of this hypothesis).

The algorithm which we sketch in this section does not employ grammar compilation. Instead, the parser uses the local structures defined in the grammar to compute a connection path from the current word to the left context *on-line*[2]. The parser described in this section is an extended version of the model described in Sturt and Crocker (1996), and further details can be found in Sturt (in preparation). The algorithm is

[2] See Konieczny and Hemforth (1994) for another example of a parser that computes projection paths on-line, though the issue of infinite local ambiguity is not addressed in that paper.

a psycholinguistic model, which is used for the investigation of syntactic *reanalysis*. We will see that the processing of left recursive structures is achieved through the application of a reanalysis operation in this model.

The grammar is expressed as a set of lexically anchored projection trees, which can be viewed as a generalization of DPS's introduced in the previous section, except that each projection tree in the grammar only represents the head projection of the anchor, and is only expanded on-line as required by the processor.

The parser proceeds by taking each word, retrieving its projection tree, and attempting to combine the projection with the current left context. The projection tree can be combined with the left context by three basic methods:

Left Combination The left context is attached as a left dependent of the projection tree. For example, if the left context is [$_{NP}$ John], and the projection tree is [$_{S}$ [$_{VP}$[$_{V}$ hates]]], then Left Combination can combine the two to form a new left context [$_{S}$[$_{NP}$ John] [$_{VP}$[$_{V}$ hates]]].

Right Combination The projection tree is attached as a right dependent of left context. For example, if the left context is [$_{S}$[$_{NP}$ John] [$_{VP}$[$_{V}$ hates]]], and the projection tree is [$_{NP}$ Mary], then Right Combination can combine the two to form the new left context [$_{S}$[$_{NP}$ John] [$_{VP}$[$_{V}$ hates][$_{NP}$ Mary]]]

Reanalysis The parser includes operations for reanalysing the analysis. In this paper, we will consider the *tree lowering* operation, in which a node N on the right frontier of the left context is detached, and left-combined with the projection tree. The projection tree is then right combined with the left context in the position previously occupied by N. For example, if the left context is [$_{S}$[$_{NP}$ John] [$_{VP}$[$_{V}$ knows][$_{NP}$ Mary]]], and the projection tree is [$_{S}$ [$_{VP}$ [$_{V}$ smokes]]], then *Mary* can be left combined with the projection tree, and the projection tree can be right combined with the left context, as follows: [$_{S}$[$_{NP}$ John] [$_{VP}$[$_{V}$ knows][$_{S}$[$_{NP}$ Mary][$_{VP}$ [$_{V}$ smokes]]]]].

If the projection tree cannot be combined with the left context directly, it is extended to successively higher levels of structure. As an illustration, imagine the parser has built the following left context:

[$_{S}$ John [$_{VP}$ thinks]]

Let us say that the next word is "his". This word is projected to the Det category. The parser cannot directly combine a determiner with the left context. For example, it cannot combine via right combination, because there is no position on the right frontier of the left context which licenses a determiner. So, the parser searches for a projection tree which selects a determiner to its left, and attempts to combine this projection with the left context. The NP projection tree allows a determiner as a left daughter, so now the parser attempts to combine an NP with the left context. This also fails, because *thinks* does not select for an NP. The parser attempts to extend the projection tree one level further, by searching for a projection which can take an NP to its left. The parser finds that an S projection can take an NP (subject) on its left. Moreover, an S category is selected by "thinks", so the parser hypothesises both NP and S projections, attaches the determiner to the NP projection, attaches the NP projection to the S projection, and attaches the S projection to the VP of the left context. The result is the following connected structure:

[$_{S}$ John [$_{VP}$ thinks [$_{S}$ [$_{NP}$ [$_{D}$ his][$_{N}$]] [$_{VP}$ [$_{V}$]]]]]

Clearly, the process of extending the projection tree will run into termination problems unless constrained in some way. The parser therefore uses the Minimal Recursive Structure as a ceiling for the height of any possible connection path. Thus, each time a node is added to the connection path, a check is made to ensure that its category has not yet appeared in the connection path.

Consider the following left recursive sentence:

(2) John hates his sister's boyfriend.

After "sister" has been incorporated, the left context will be as follows:

[$_{S}$ [$_{NP}$ John] [$_{VP}$ hates [$_{NP}$ his sister]]]

We assume that the particle *'s* projects to a determiner, taking an obligatory NP on its left.

[$_{D}$ NP 's]

This means that the particle cannot combine with the left context through standard left or right combination. However, it can combine via the *tree lowering* reanalysis operation (described above). The NP dominating "his sister" is detached from its position as the argument of "hates",

[$_{S}$ [$_{NP}$ John] [$_{VP}$ hates .NP]] [$_{NP}$ his sister] [$_{D}$ NP 's]

and left-combined as the argument of the possessive particle.

[$_{S}$ [$_{NP}$ John] [$_{VP}$ hates .NP]] [$_{D}$ [$_{NP}$ his sister] 's]

The Det projection of the possessive particle cannot attach directly to the left context, but by projecting to the level of NP, it can.

[$_{S}$ [$_{NP}$ John] [$_{VP}$ hates .NP]] [$_{NP}$ [$_{D}$ [$_{NP}$ his sister] 's] .n]

The resulting left context will then look like this:

[$_{S}$ [$_{NP}$ John] [$_{VP}$ hates [$_{NP}$ [$_{D}$ [$_{NP}$ his sister] 's] .n]]]

When "boyfriend" is read, it is attached as the head of the newly created NP.

[$_{S}$ [$_{NP}$ John] [$_{VP}$ hates [$_{NP}$ [$_{D}$ [$_{NP}$ his sister] 's]

boyfriend]]]

The left recursive structure could potentially be extended in this way indefinitely, with a new instantiation of minimal recursive structure inserted into the right frontier each time the possessive particle is read.

(3) John hates his sister's boyfriend's mother's psychoanalyst's .....

The parser can be viewed as a bottom-up algorithm with a predictive component, which maintains a permanently connected structure, allowing for strong incrementality. The building of structure is triggered by lexical input, but driven by the need for connectedness, and constrained by minimal recursive structure. The algorithm also satisfies the criteria proposed by Abney and Johnson (1991) with respect to space requirements and local ambiguity.

The basic mechanism described here has been used to investigate psycholinguistic preferences for reanalysis in Japanese (see Sturt and Crocker (1996)), where infinite local ambiguity of the variety we have been discussing in this paper abounds.

Conclusions

This paper has discussed the problem of infinite local ambiguity, which must be addressed by any strongly incremental parsing model. We have argued that a successful solution to this problem must employ the notion of minimal recursive structure, and we have presented two algorithms which use this notion in order to combine strongly incremental processing with the successful handling of infinite local ambiguity.

The method has been tested on context free grammar, which is the basis for a number of natural language formalisms. The extension to these more powerful formalisms is currently under study. In order to use the method with wide coverage grammars, we are also investigating the use of graph structures for packing DRS's which share syntactic structure.

References

Abney S. P., A computational Model of Human Parsing. *Journal of Psycholinguistic Research*, Vol. 18, N. 1, 1989, pp. 129-144.

Abney S. P. and M. Johnson, Memory Requirements and Local Ambiguities of Parsing Strategies. *Journal of Psycholinguistic Research*, Vol. 20, N. 3, 1991, pp. 233-250.

Bader, M. and I. Lasser. 1994. German verb-final clauses and sentence processing. In C. Clifton, L. Frazier, and K. Rayner, editors, *Perspectives on Sentence Processing*. Lawrence Erlbaum Associates, New Jersey.

Berwick, R.C. and Weinberg, A.S. 1985. Deterministic parsing and linguistic explanation. *Language and Cognitive Processes*, 1, pp. 109-134.

Earley, J. 1970. An efficient context-free parsing algorithm. *Communications of the Association ofr Computing Machinery*, 1, pp. 94-102.

Eberhard K., Spivey-Knowlton M., Sedivy J., Tanenhaus N., Eye movements as a window into real-time spoken language comprehension in natural contexts, *Journal of Psycholinguistic Research*, 24, 1995, pp. 409-436.

Frazier, L. 1978. *On comprehending sentences: Syntactic parsing strategies*. Ph.D. thesis, University of Connecticut.

Frazier, L. and K. Rayner. 1988. Parameterizing the language processing system: Left-vs. right-branching within and across languages. In J.A. Hawkins, editor, *Explaining Language Universals*. Basil Blackwell, pages 247-279.

Gibson, E. 1991. *A Computational Theory of Human Linguistic Processing: Memory Limitations and Processing breakdown*. Ph.D. thesis, Carnegie Mellon University.

Gorrell, P. 1995. *Syntax and Parsing*. Cambridge: Cambridge University Press.

Hemforth, B., L. Konieczny, and C. Scheepers. 1994. On reanalysis: Head position and syntactic complexity. In B. Hemforth, L. Konieczny, C. Scheepers, and G. Strube, editors, *IIG-Berichte* 8/94. Institut für Informatik und Gesellschaft, IIG, University of Freiburg, pages 23-50.

Kay, M. 1980. *Algorithm Schemata and Data Structures in Syntactic Processing*. No. CSL-80-12. XEROX PARC, Palo Alto, CA.

Konieczny, L. and B. Hemforth . 1994. Incremental parsing with lexicalized grammars. pp. 113-133. In B. Hemforth, L. Konieczny, C. Scheepers, and G. Strube, editors, *IIG-Berichte* 8/94. Institut für Informatik und Gesellschaft, IIG, University of Freiburg, pages 23-50.

Lombardo, V. 1995. Parsing and Recovery. In *Proceedings of the XVII Annual Conference of the Cognitive Science Society*.

Lombardo, V. 1996. A Computational Model of Recovery. To appear in J.D. Fodor and F. Ferreira (eds.), *Reanalysis in Sentence Processing, Kluwer Academic Publishers*.

MacDonald, M.C., N.J. Pearlmutter, and M.S. Seidenberg. 1994. Lexical nature of syntactic ambiguity resolution. *Psychological Review*, 101(4):676-703.

Marslen-Wilson, W. 1975. Sentence perception as an interactive parallel process. *Science*, 189:226-228.

Mazuka, R., K. Itoh, S. Kiritani, S. Niwa, K. Ikejiri and K. Naitoh. 1989. Processing of Japanese garden-path, center-embedded, and multiply-left-embedded sentences: Reading time data from an eye movement study. *Annual Bulletin of the Research Institute of Logopedics and Phoniatrics*, 23, pp. 187-212

Milward, D. 1995. Incremental Interpretation of Categorial Grammar. In *Proceedings of the 7th conference of the European chapter of the Association for Computational Linguistics*, p.119-126

Pritchett, B.L. 1992. *Grammatical Competence and Parsing Performance*. Chicago, IL: University of Chicago Press.

Sturt, P. in preparation:. *Syntactic Reanalysis in Human Language Processing*. Ph.D. thesis, Centre for Cognitive Science, University of Edinburgh, Edinburgh, Scotland.

Sturt, P. and M.W. Crocker. 1996. Monotonic syntactic processing: a cross-linguistic study of attachment and reanalysis. *Language and Cognitive Processes*, 11(5):449-494.

Tomita M., An Efficient Augmented-Context-Free Parsing Algorithm, *Computational Linguistics* 13, 1987, pp. 31-46.

Thompson H.S., M. Dixon and J. Lamping. Compose-Reduce Parsing. In *Proceedings of the 29th Meeting of the Association for Computational Linguistics*, pp. 87-97.

Yamashita, K. 1994. *Processing of Japanese and Korean*. Ph.D. thesis, Ohio State University, Columbus, Ohio.