COMPUTATIONAL MODELS OF SYNTACTIC PROCESSING IN HUMAN LANGUAGE COMPREHENSION

Gerard Kempen
Experimental and Theoretical Psychology Unit
University of Leiden

November 1994

Published in 1996 as Chapter 8 in: In: Dijkstra, T. & De Smedt, K. (Eds.), *Computational psycholinguistics: symbolic and subsymbolic models of language processing*. London: Taylor & Francis. [pp. 192-220]

8.1 INTRODUCTION

The pioneers of computational psycholinguistics, who built their earliest models more than twenty years ago, were primarily interested in syntax. Novel computational techniques for analyzing the structure of natural language sentences inspired them to develop parsers that could simulate characteristic phenomena of human sentence processing. Favorite objects of study were *garden-path sentences* that would lead us up the garden path due to strong interpretive biases. In 1972, R. Kaplan introduced *Augmented transition networks as psychological models of sentence comprehension*. At about the same time, J. Kimball (1973) proposed *Seven principles of surface structure parsing in natural language* which have inspired psycholinguistic parser design for many years.

This chapter takes stock of what has happened since. Section 8.2 summarizes the results of experimental psycholinguistic work on the syntactic aspects of sentence comprehension (leaving aside semantic issues). Then, in Section 8.3, I outline five simulation models of syntactic parsing which have attempted to take such findings into account. Section 8.4 presents some conclusions and tries to catch a glimpse of the future.

8.2 THE EMPIRICAL ARENA

In this Section I present five groups of syntactic processing phenomena that have been established experimentally. Due to limitations of space I cannot go into procedural details of the experiments nor, with a few exceptions, into the theoretical debates that have accompanied the interpretation of certain sets of data. I refer to Mitchell (1994) for an extensive survey of the empirical literature. I will concentrate on clear cases—those areas in the arena where the dust has settled or is settling.

My way of grouping the data and labeling the groups is not entirely conventional. The particular headings I have chosen were guided by the following hypothesis: Processes and mechanisms known to be operative in other cognitive domains, also underlie linguistic performance. The empirical phenomena reflect the way these *dynamic* factors cope with linguistic *structures* of various kinds and varying levels of complexity. This is why I have foregrounded such notions as frequency, priming, recency, control, and capacity at the expense of concepts that denote syntactic configurations such as Minimal Attachment, filler-gap relationship, or various types of clause embedding.

8.2.1 Process control

Bounded parallelism. The human syntactic processor does not compute all syntactic structures allowed by an ambiguous input sentence. On the contrary, there is overwhelming evidence that, in the course of processing *structurally ambiguous* sentences, no more than one syntactic representation is constructed. Consider the examples of structural syntactic ambiguity in (1) and (2).

(1) John bought the book that I had been trying to obtain for Susan.

(2) Welke dokter heeft deze patient bezocht?

(a) Which doctor has this patient visited?

(b) Which doctor has visited this patient?

Few readers of (1), for example, will spontaneously realize that *for Susan* can be interpreted as modifying *bought*. And speakers of Dutch consider only one analysis of sentence (2)—usually (b) where the *Wh*-phrase serves as grammatical subject (Frazier & Flores d'Arcais 1989). On-line measures of processing load have failed to support the hypothesis that readers or listeners construct and maintain multiple analyses while comprehending such sentences.

Structural syntactic ambiguity originates from the fact that certain constituents may contract several different grammatical relationships. It is to be distinguished from *lexical frame ambiguity* as illustrated by examples (3) and (4). Compare the short version of the sentences (without bracketed string) with the long one.

(3) Sally found out the answer to the difficult physics problem [was in the book].

(4) The soldiers warned about the dangers [conducted the midnight raid].

As these example show, verbs may belong to different *subcategorizations* according to their environments. The lexical item *find out* has two different lexical frames: it can take a noun phrase (NP) as direct object or a complement clause. And *warned* is ambiguous between finite verb and past participle. Gorrell (1991) and MacDonald, Just and Carpenter (1992) obtained evidence for parallel processing. Their data suggest that the alternative lexical frames are both activated and maintained while the ambiguous region is being processed. This observation is in keeping with what is known about processing *semantically* ambiguous lexical items. Multiple meanings are activated initially, but only the contextually appropriate one is retained (Swinney 1979; Rayner, Pacht & Duffy 1994).

To sum up this point, the human syntactic processor avoids the construction of multiple syntactic analyses for complete sentences, although it seems to consider the alternatives offered by syntactically ambiguous lexical items. The upshot is parallelism of a very restricted kind (see Garrett, 1990 for a similar view). At the end of Section 8.3.5, I will return to the issue of parallel computing.

Immediacy. In addition to the options discussed so far (namely, "single-track" and "multiple-track" processing), there is a third possibility for dealing with syntactic ambiguity. When encountering a syntactically ambiguous region, the parser might adopt a superficial style of analysis and only commit itself to structural decisions that are neutral with respect to the alternative solutions: "minimal commitment parsing". However, the on-line processing data are at variance with this proposal, as argued at length by Mitchell (1994); see also Frazier and Flores d'Arcais (1989) for an empirical argument based on Dutch. The human syntactic processor often makes strong commitments without delay. A clear example is provided by the

Active Filler Strategy which is responsible for a temporary trouble spot right after *forced* in (5) (cf. Fodor, 1989 and Pickering & Barry, 1991).

(5) Who could the little child have forced us to sing those stupid French songs for, last Christmas?

The parser needs to find out which lexical frame the preposed *who* belongs to. The first candidate is the direct object slot of *force*. The presence of *us* prevents this, thereby causing a processing problem. This observation suggests that the parser indeed attempts to drop the interrogative pronoun at the first location that seems suitable. The next slot, the one provided by *for*, brings success (compare *For whom could the little child …*).

Incremental and interactive processing. That language utterances are analyzed syntactically from left to right on a word-by-word basis (rather than, e.g., clause-by-clause) is commonly assumed: *incremental* syntactic processing. Somewhat controversial, though widely accepted, is the further assumption that this characterization also applies to the semantic interpretation of utterances. That is, the unfolding semantic representation is updated at every new content word appearing in the input.

The incremental mode of syntactic analysis and semantic interpretation creates opportunities for *semantic-syntactic interactions*: semantic decisions taken earlier on in the sentence or in previous context sentences may affect subsequent syntactic choices. Under what conditions such influences make themselves felt, and how strong they are, are hotly debated and intensely researched issues. A broad spectrum of opinions has been voiced, based on sometimes conflicting empirical evidence. (For some highlights in the literature, I refer to Frazier, 1978; Crain & Steedman, 1985; Taraban & McClelland, 1988; Rayner, Garrod & Perfetti, 1992; Altmann, Garnham & Dennis, 1992; Britt, 1994; and Trueswell, Tanenhaus & Garnsey, 1994.) What has become clear is that the human syntactic processor is not immune to semantic (conceptual, pragmatic, contextual) factors, rendering the position of full "autonomy of syntax" untenable. However, at the current state of play it is impossible to draw the contours, let alone the detailed shape, of an empirically well-grounded interactive model.

8.2.2 Lexical frame preferences

Lexical frames (or subcategorization frames) express constraints on the shape of possible grammatical environments of a word. This information, which belongs to that word's entry (*lemma;* see Chapter 12) in the mental lexicon, specifies grammatical properties of optional or obligatory complements and modifiers. In the context of examples (3) and (4) above we have already seen that words may have more than one lexical frame associated with them. Moreover, the evidence for bounded parallelism suggests that several lexical frames of a word may be simultaneously active. This does not imply, however, that the syntactic processor treats the alternative frames on equal footing. One of the causes of *garden-pathing* has to do with preference for one lexical frame over another. Such preferences can be assessed in various ways, e.g. in sentence production tasks where subjects are given a verb and generate a sentence containing it. Clifton, Frazier and Connine (1984) compared sentences like those in (9).

(9) a. The aging pianist taught *his* solo with great dignity.

b. The aging pianist taught *with* his entire family.

c. The aging pianist performed *his* solo with great dignity.

d. The aging pianist performed *with* his entire family.

The verbs *teach* and *perform* have similar lexical frames in that they both allow a transitive or an intransitive frame. However, while *teach* prefers a direct object, *perform* is preferably used intransitively. The underlined words mark the onset of the disambiguating region: from that position onwards, it is clear which of the two frames has to be instantiated. Subjects were presented with such sentences in a word-by-word reading task. Immediately after presentation of the disambiguating word they carried out a visual lexical decision task. Reaction times were longer in sentences with unpreferred lexical frames, i.e. in (9b) and (9c). Giving up the preferred frame apparently increased processing load. A recent study by Shapiro, Nagel and Levine (1993) corroborates this result.

Lexical frame preferences have played an important role in a controversy between *phrase-structure driven* and *lexically driven* models of parsing. The former models hold that the parser constructs an initial syntactic tree on the basis of syntactic phrase-structure rules, guided by parsing strategies such as and Right Association and Minimal Attachment. (These concepts are explained in Sections 8.2.3 and 8.3.2, respectively.) Lexical frames are consulted at a later stage and help to confirm or to improve the earlier phrase-structure decisions. In lexically driven models, on the other hand, frame information guides phrase-structure decisions right from the start. They are probably in better agreement with empirical evidence, a few stubborn data notwithstanding (Mitchell, 1987).

8.2.3 Syntactic frequency

Cuetos and Mitchell (1988) discovered that Spanish readers have a different preference for attaching relative clauses than English readers. The sentences they presented to subjects contained a complex NP with two possible attachment points for a final relative clause. They showed that, while English readers prefer the low attachment point with the relative clause in (10a) modifying *actress*, Spanish readers prefer high attachment with *hermano* modified by the relative clause in (10b). The relative clause does not contain any gender clues.

(10) a. Someone shot the brother of the actress who was on the balcony.

b. Alguien disparó contra el hermano de la actriz que estaba en el balcón.

c. Alguien disparó contra el hermano de la actriz que estaba en el balcón *con su marido*.

d. Alguien disparó contra *la hermana* de la actriz que estaba en el balcón *con su marido*.

In a self-paced reading task, this preference showed up in the viewing times of the passage *con su marido* (*with her husband*) in (10c). These viewing times were longer than in a control condition with *el hermano* replaced by *la hermana* (sister) in (10d). In subsequent experiments it could be verified that the attachment preference indeed reflects the frequency of the two NP constructions in Spanish. Additional correspondences between syntactic frequency and parsing preference have been observed for other constructions and other languages (Gibson & Loomis, 1994; Cuetos, Mitchell, & Corley, in press).

8.2.4 Temporal effects

Syntactic priming. Facilitation occurs when two or more similar syntactic constructions that share a problematic feature are processed in close temporal succession. Frazier, Taft, Roeper, Clifton, and Ehrlich (1984) demonstrated this for coordinate structures exemplified by (11) and (12).

(11) Jim believed all Tom's stories [were literally true] and Sue believed Jim's stories [were fictitious].

(12) Mary wrote a long note about her predicament to her mother and Sue wired to her father a telegram requesting more money.

In various trials of a self-paced reading experiment, the bracketed strings of (11) could be present or absent. Without the strings, *Tom's stories* is direct object of *believe*; in the long version with the strings added, this NP is the subject of the embedded complement clause. The long version of the second conjunct was significantly easier to process when the first member was also presented in the long version. No signs of garden-pathing remained. A similar effect obtained when both conjuncts contained a *heavy NP shift* causing the shorter indirect object to precede the longer direct object (cf. Chapter 11). So, (12) was more difficult than its counterpart with *to her mother* immediately following *wrote*. Schuster and MacDonald (1994) have obtained syntactic priming effects in texts where the priming and the primed constructions occurred in different sentences.

Syntactic recency. The best known and least controversial phenomenon of human syntactic parsing was termed Right Association by Kimball (1973). One of his examples is (13).

(13) Joe said that Martha expected that it would rain yesterday.

(14) Since Jay always jogs a mile seems like a short distance to him.

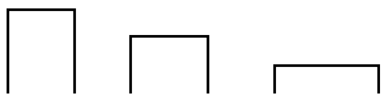
The principle predicts that a new constituent will preferably be attached to the lowest possible (most recent, rightmost) non-terminal node of the current syntactic tree. The most likely attachment point for the adverbial phrase (AP) *yesterday* in (13) will be the verb phrase (VP) dominating *rain*. Readings with the AP associated with *expect* and *say* will be increasingly difficult. The principle also predicts that *a mile* in (14) will initially be analyzed as belonging to the subordinate rather than to the matrix (main) clause. Other names for the same or very similar principles are Late Closure (Frazier & Rayner, 1982) and Recency Preference (Gibson, 1990).

Constituent length effects. One of the experimental manipulations in Frazier and Rayner's (1982) classical eye-tracking study concerned the length of the ambiguous phrase in garden-path sentences. The sentence material contained temporary object-subject ambiguities in preposed subordinate clauses, e.g. (14) and in object complement clauses, e.g. (3). For example, in the short version parallel to (3), the ambiguous region *the answer to the difficult physics problem* was shortened to *the answer*; and in the long version of (14), *a mile* had been replaced by *a mile and a half.*

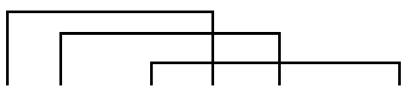
The eye movement data clearly showed that after a long ambiguous region the garden-path effect was larger than after a short one. The short versions of (3) produced only weak signs of garden-pathing. This indicates that ongoing processing is interrupted only beyond a minimum length of the ambiguous region. Another effect of length is discussed in Section 8.3.5 below in the context of examples (21a,b).

8.2.5 Working memory capacity

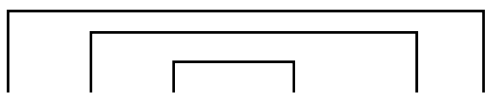
The computations performed by the human syntactic processor take place in some working memory—a device for short-term storage of lexical, grammatical and semantic information retrieved from long-term memory or generated in the course of linguistic processing. Capacity limitations of this workspace are often held responsible for language comprehension problems, e.g. when sentences become excessively long or when subject and finite verb are far apart, as easily happens in German and Dutch subordinate clauses. Under this heading I also list the notorious doubly center-embedded clauses, e.g. (15c), that are so much harder to understand than the unproblematic single center-embeddings. Remarkable also are the contrasts between center-, cross-serial and righthand embeddings, whose comprehensibility increases in this order. Bach, Brown and Marslen-Wilson (1986) have shown that this rank order applies cross-linguistically:



(15) a. … that John saw Peter help Mary to swim. (Righthand/English—Easy)



b. … dat Jan Peter Marie zag helpen zwemmen (Cross-serial/Dutch—Intermediate)



c. … daß Johann Peter Maria schwimmen helfen sah. (Center/German—Hard)

8.3 THE SIMULATION AMPHITHEATER

8.3.1 Preliminaries

For over twenty years, theory- and computer-minded onlookers—a colorful mixture of psychologists, linguists and computer scientists—have been watching the events in the empirical arena, comfortably seated in the amphitheater. Only few of them have descended into the arena to intervene in the spectacle, while the experimental psycholinguists knew they were the stars and most of them preferred to keep it that way. The upshot is that, as we will see, the simulation models do not always bear a close relationship to the empirical facts, and that the course of experimental work has hardly been influenced by results of computer modeling studies.

In this section, I will present five different computational models explicitly aiming at psychological plausibility. They all try to simulate some selection of phenomena from the empirical arena. The models are

1. Augmented Transition Networks (ATNs; Kaplan, 1972, 1975)
2. Shift-Reduce Parsing (Shieber, 1983, Pereira, 1985, Abney, 1989)
3. PARSIFAL (Marcus, 1980)
4. Race-Based Parsing (McRoy & Hirst, 1990), and
5. Unification Space (U-Space; Kempen & Vosse, 1989).

ATNs and PARSIFAL have been widely discussed in the literature. Shift-Reduce Parsing is less known in psycholinguistic circles but helps to explain essential properties of PARSIFAL. Race-Based Parsing was inspired by probably the best known non-computational model of the human syntactic processor: Frazier and Fodor's (1978) SAUSAGE MACHINE. The first four models are strictly symbolic (in the sense explained in Chapter 2); the fifth model is *hybrid* in that it mixes symbolic and connectionist features.

Table 8.1. Sample grammar associated with sentences (16a/b).

1. S → NP VP	6. RRC → Vpass NP	11. PP → Prep NP	16. N → letter
2. NP → Art N	7. VP → Vintr	12. Vintr → fainted	17. N → student
3. NP → PropN	8. VP → Vintr PP	13. Vtr → read	18. PropN → Chrysanne
4. NP → NP PP	9. VP → Vtr NP	14. Vpass → read	19. Prep → to
5. NP → NP RRC	10. VP → Vtr NP PP	15. Art → the	

In order to enhance comparability of the models I will often use the simple context-free grammar of Table 8.1 to explain the essence of their inner workings. This grammar generates, among other things, the following sentences in (16), adapted from McRoy and Hirst (1990).

(16) a. The student read the letter to Chrysanne.[i]

[i] This sentence is ambiguous: *to Chrysanne* can modify either *read* or *the letter*.

b. The student read the letter to Chrysanne fainted.[ii]

The basic architecture of any syntactic parser is depicted in Figure 8.1. The processor-cum-working memory reads words from the input buffer, consults the lexical, morphological and syntactic information associated with them, and assembles a syntactic structure as output—a complete or fragmentary syntactic tree. The details vary between models, of course. The box drawn with dotted lines denotes an optional semantic/pragmatic component which codetermines the parsing process in parallel with lexical/grammatical information. This would constitute an interactive model. In a syntax-first model, the semantic/pragmatic component would operate upon syntactic structures tentatively proposed by the syntactic processor (serial connection and feedback). In the context of *human* syntactic processing one can safely assume that the grammatical, lexical, and large parts of the conceptual knowledge are stored in long-term memory.

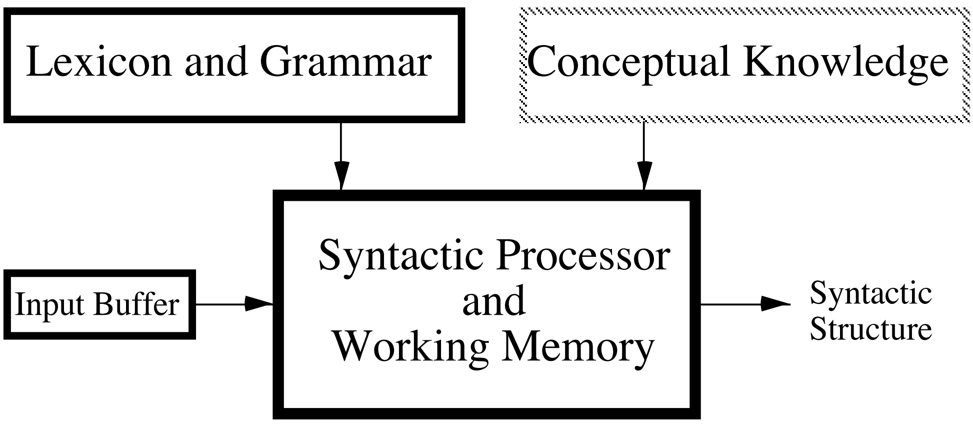


Figure 8.1. Main parser components.

8.3.2 Augmented Transition Networks

In an ATN, the parsing process is represented as a collection of transitions between states (see Chapter 2). The states are usually depicted as labeled nodes, the transitions as directed labeled arcs between nodes. Initial and final states are connected via any number of intermediate states and arcs. Graphs representing the set of permitted transitions embody grammar rules. Words in the input are consumed one by one. Their properties determine how a graph can be traversed. If more than one arc leaves a state, they are tried out sequentially in clockwise direction. An ATN corresponding to the grammar of Table 8.1 is depicted in Figure 8.2.

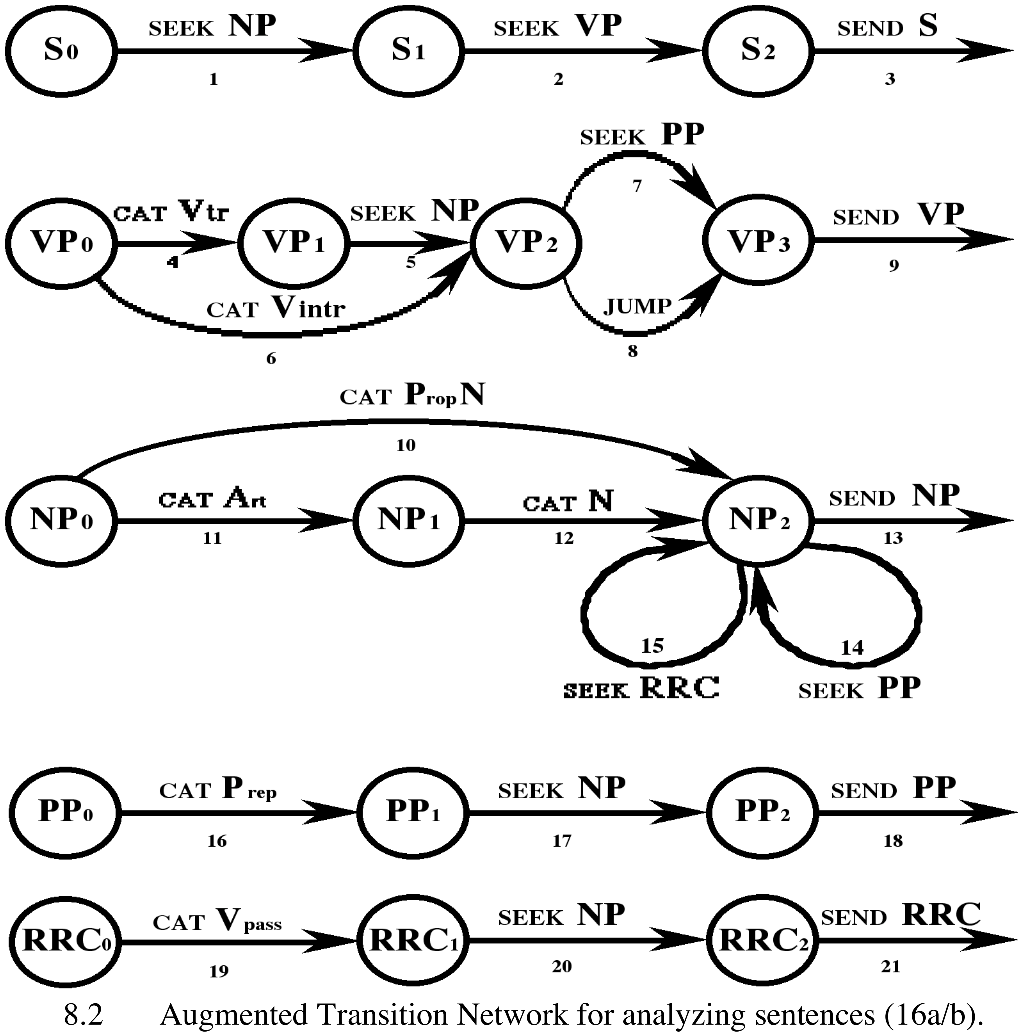
For each of the five phrasal categories S (sentence), NP (noun phrase), VP (verb phrase), RRC (reduced relative clause) and PP (prepositional phrase) there are separate networks which are specialized in parsing Ss, NPs, VPs, RRCs, and PPs. A string is accepted as a sentence if it can

---

[ii] The verb *read* in (20b) opens a reduced relative clause (RRC; cf. *The student who was read the letter, fainted*). I disregard the semantically odd analysis of *the letter to Chrysanne fainted* as an object complement clause (*The student read that the letter to Chrysanne fainted*).

move the top ATN from state S_0 to S_2. A string that can bring the NP network from NP_0 to NP_2 is a noun phrase.

Arc labels denote condition-action pairs (printed above the arc). CAT arcs specify which word category the next input item should belong to (e.g., Article, Proper Noun, Transitive verb). SEEK arcs are calls to another subnetwork. For instance, SEEK NP causes transfer of control to the NP subnetwork, which then undertakes to interpret (part of) the remaining input as an NP. SEND arcs return control to the "calling" subnetwork and arc. For example, the S subnetwork opens with a call to the NP subnetwork. If this indeed can recognize an NP in the first part of the input string, then the SEND NP arc leaving state NP_3 transfers control to the S subnetwork, which then moves to state S_1. The last type of arc label (JUMP) indicates that the arc may be traversed for free, without any conditions imposed. For instance, the JUMP arc in the VP subnetwork renders the PP optional. The numbers below the arcs refer to a (possibly empty) list of actions to be executed when the arc is traversed—usually bookkeeping operations needed to put together part of a tree. For instance, Action 1 in the S-network assigns the role of Subject to the recognized NP. An excellent source of further information is Winograd (1983



8.2 Augmented Transition Network for analyzing sentences (16a/b).

How does this ATN fare with sentence (16b)? Would it yield a garden-path effect? Let us consider Figure 8.3a, which shows the arcs traversed until hitting upon the final word *fainted*.

At that point a Sentence has been accepted although the input has not yet finished. This induces a rather complex *backtracking* operation (this concept is explained in Chapter 2). It turns out that a wrong path was taken almost at the beginning. If, within the first instantiation of the NP-network, after *student*, arc 15 had been chosen instead of arc 13, everything would have fallen out well. This is illustrated in Figure 8.3b.

A

```
(1                     2                                                      3)
 (11-the 12-student 13) (4-read 5                     7                      9)
                                 (11-the 12-letter 13) (16-to 17           18)
                                                               (10-Chrys. 13)
```

B

```
(1                                                                          2              3)
 (11-the 12-student 15                                                   13) (6-fainted 8 9)
                     (19-read 20                                        21)
                               (11-the 12-letter 14                    13)
                                                  (16-to 17           18)
                                                          (10-Chrys. 13)
```

Figure 8.3. ATN parsing of sentence (16b). Backtracking operations are not shown.

The reader who handparses the sentence will notice that it is the ordering of outgoing arcs which causes the garden-path. Simply scheduling both SEEK arcs leaving NP_2 before the Send arc will pre-empt the need to backtrack here. But of course, in that case sentence (16a), without *fainted*, becomes problematic.

In a 1980 paper, Wanner describes a general arc ordering scheme intended to account for two interpretation biases that in those days were hardly disputed: Minimal Attachment and Right Association (see also Frazier & Fodor, 1978 and Fodor & Frazier, 1980). Right Association has survived until the present day (in Section 8.2.4 it was rechristened Syntactic Recency). The Minimal Attachment principle predicts that in case of syntactic ambiguity the structure with fewer nodes will be preferred. Sentence (16a) provides an illustration. Figure 8.4a shows the Minimal Attachment analysis, with the PP attached to the VP. In Figure 8.4b, the PP modifies the object NP. The latter analysis requires one more node, thus violating Minimal Attachment. The principle also accounts for the garden-path character of (16b). Figure 8.4c shows that analyzing (16a) as a Noun Phrase, i.e. as the subject of (16b), also costs one more node than the Minimal Attachment analysis in Figure 8.4a.

Wanner proposes to schedule CAT arcs before SEEK arcs, and both of these before JUMP and SEND arcs. (JUMP and SEND arcs need not be ordered because Wanner assumes they never leave the same state.) Modifying Figure 8.2 accordingly, i.e., scheduling SEEK arcs 14 and 15 in the NP subnetwork before SEND arc 13, indeed yields Right Association: *to Chrysanne* will become attached to the NP rather than to the VP (*letter to Chrysanne* instead of *read ... to Chrysanne*). However, Wanner's proposal cannot manage with Minimal Attachment. The problem is rooted in the presence of *left-recursive* rules in the sample grammar, i.e., rewrite rules where the first symbol at the righthand side is identical to the lefthand symbol[iii].

[iii] For a more general definition of left-recursion see, e.g., Aho & Ullman (1972).

Examples are rules 4 and 5 in Table 8.1: *NP* → *NP PP* and *NP* → *NP RRC*. A straightforward conversion of rule 4 leads to the network of Figure 8.5. This ATN, however, gets easily trapped in infinite recursion. Consider a parser that consists of the ATN of Figure 8.2 with the NP subnetwork replaced by Figure 8.5. How will it deal with garden-path sentence (16c)?

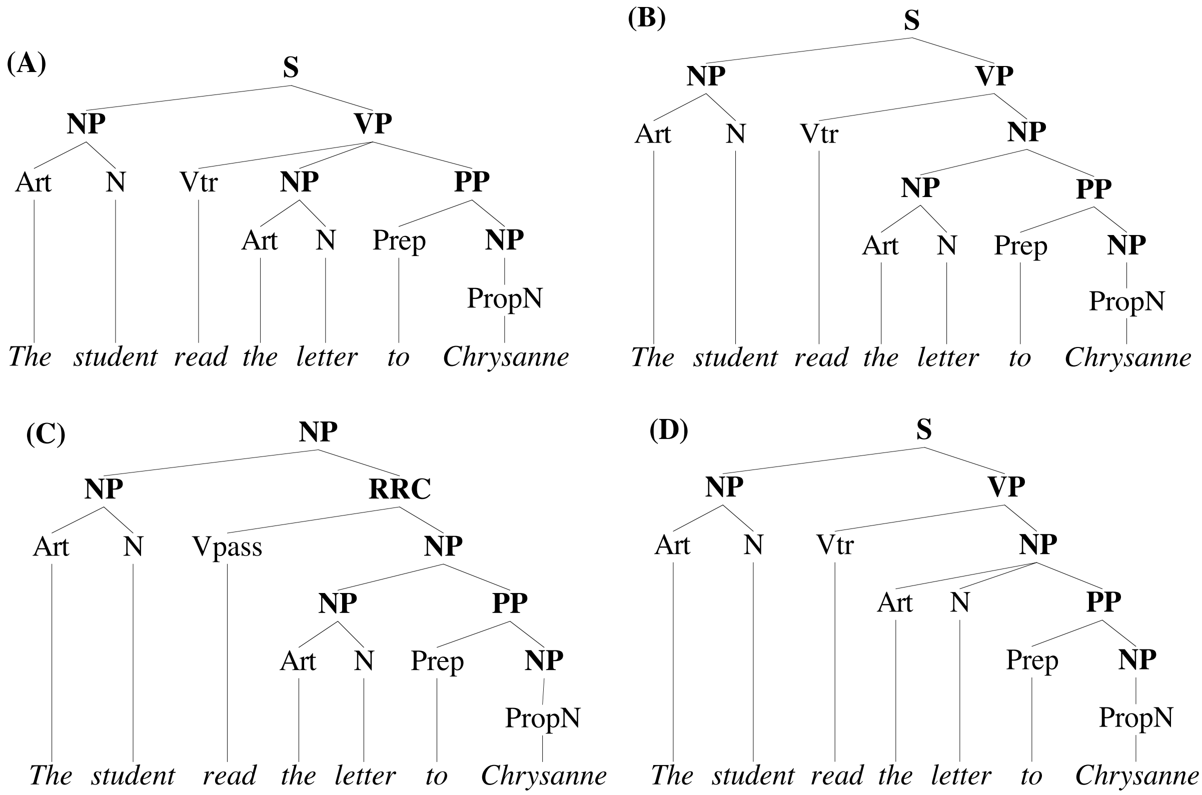


Figure 8.4. Minimal Attachment (a) and Non-Minimal Attachment (b, c) analyses of (16a) according to the original sample grammar of Table 8.1. The analysis shown in (d) is based on a modified sample grammar without left-recursive rules.

(16c) The student read the letter fainted.

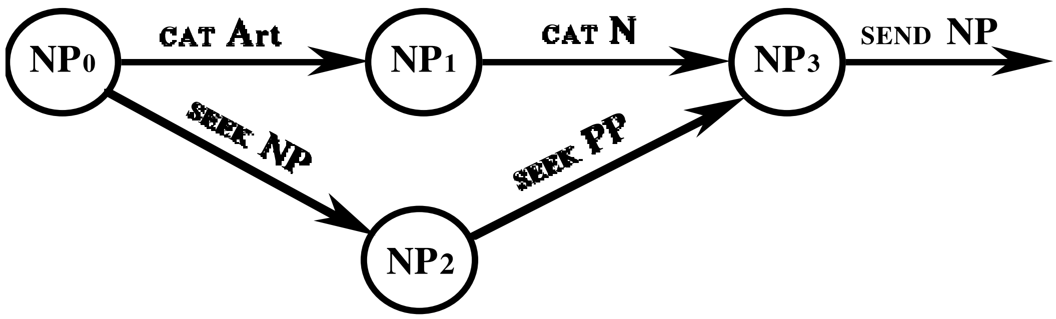


Figure 8.5. ATN with left-recursion (cf. rule 4 of Table 8.1).

After the VP subnetwork has identified *the letter* as direct object NP, the verb *fainted* poses a problem. This triggers a backtracking operation, causing the processor to traverse the SEEK NP arc that leaves NP_0. A new instantiation of the NP net is created, which again interprets *the* *letter* as an NP. The parser, now in state NP_2, cannot proceed any further since *fainted* is not a preposition. Backtracking to state NP_0 now triggers another instantiation of the NP net, which falls into the same trap as its predecessor—and so on indefinitely.

The lesson to be learned from this example is to avoid left-recursion in ATNs (see also Winograd, 1983). The NP network depicted in Figure 8.2, which does not suffer from this problem, has another drawback though. It does not match rules 4 and 5 of the sample grammar. Instead, it corresponds to non-left-recursive rules (4') *NP* → *Art N PP*$^+$ and (5') *NP* → *Art N RRC*$^+$. (The plus sign indicates that the constituent involved may occur more than once.) Now consider the alternative sample grammar that results from replacing the original rules 4 and 5 by 4' and 5'. The Minimal Attachment principle no longer predicts a preference for interpreting *to Chrysanne* as a VP rather than as an NP modifier. The two attachments cost the same number of nodes: compare Figures 8.4a and 8.4d.

The conclusion must be that Wanner's proposal cannot provide a viable account of Minimal Attachment preferences. However, from a present-day perspective this is not a serious disadvantage. Already for some time the principle is under heavy fire from experimental corners (see Mitchell, 1994). The experimental results it is supposed to cover were sometimes difficult to replicate, or can be explained on different grounds (e.g. Lexical Frame Preferences or Syntactic Frequency; cf. Sections 8.2.2 and 8.2.3). The odds are that the Minimal Attachment Principle will be abandoned and its duty taken over by a combination of dynamic factors.

Successful attempts to confront ATNs with a more encompassing set of empirical data have not been reported in the literature. Probably the biggest disadvantage of ATNs is their rigid top-down control structure which leaves little room for typically bottom-up phenomena such as Lexical Frame Preferences or Syntactic Priming. Bottom-up parsing techniques are clearly gaining the upper hand.

8.3.3 Shift-Reduce Parsing

The technique of Shift-Reduce parsing has been developed by computer scientists in the context of parsing programming language expressions. The basic ingredients are three data structures: *input buffer*, *push-down stack*, and *control table*. The input buffer contains the string of words to be processed and is scanned from left to right.. The stack is a linear arrangement of symbols, initially empty, which functions as a working memory for saving intermediate results. The control table is a procedural embedding of the grammar rules. It specifies which actions must be taken when certain syntactic conditions are met.

The actions decreed by the control table are of two kinds: *shifts* and *reductions*. A shift is executed by removing the leftmost word waiting in the buffer and placing it on top of the stack, thereby pushing down any other symbols residing on the stack. A reduction is like applying a grammar rule in reverse. For example, instead of *rewriting* the symbol NP as Art N in accordance with the second rule of the sample grammar, we *reduce* the sequence Art N to NP. More precisely, if the topmost stack symbol is N and the one underneath Art, then these two are removed and replaced by the single symbol NP—the new top symbol on the stack. In terms of the rewrite rules in Table 8.1, a reduction causes one or more stack symbols which correspond with the right-hand side of a rewrite rule, to be substituted by the left-hand side of that rule. It will be clear that a reduction is a bottom-up operation.

In LR(1) parsing, a special variety of Shift-Reduce parsing, undigested input words waiting in the buffer may be referenced in addition to stack symbols. This constitutes a kind of look-ahead which helps to select the best possible next action. In LR(1) parsing, only the first word waiting in the input buffer may be inspected.

At any point in time, the fact that certain relevant events have occurred earlier during the parse is encoded by the current parser state. States are pushed onto the stack, just like the input words. The most recently assigned state on the stack counts as the current state. The action prescribed by the control table is determined by three conditions: the current state, the stack top symbol and (possibly) the look-ahead item.

Table 8.2 is a control table corresponding to the sample grammar in Table 8.1.[iv] In the lefmost column, the possible states of the parser are designated by integer numbers (0-25). Their order is of no particular importance.[v] The next 12 columns (Art - Vtr) refer to the topmost symbol on the stack, the 8 rightmost columns to the first word in the input buffer (# is end of input). Cell contents indicate actions to be performed. Numbers indicate the transition from one state to the next. For example, if the parser is in state 0 and an Art is the topmost symbol on the stack, the current state is changed to 1. Syntactic categories in the cells prescribe reductions and Sh indicates a shift. If a cell contains two lines, there is a reduce-reduce or shift-reduce conflict. The dagger in row 4 means *accept*.

Table 8.3 shows how the LR(1) parser deals with example (16b). Each row corresponds to a step in the parsing process. The leftmost column represents the stack, with the top pointing to the right. The middle column shows the first few undigested words in the input buffer waiting to be processed. The rows of the right-hand column specify the action dictated by the control table in response to the configuration of stack top, stack state and look-ahead in the same row. For instance, the first row shows the parser in its initial state (#0), with an empty stack (Ø). State #0 together with the first buffer item triggers action "Shift, →6". (See the first cell in the *the*-column of Table 8.2. The triggering elements of the current configuration are underlined in Table 8.3.) The processor eliminates *the* from the buffer, pushes it onto the stack, and enters state #6. The effects of this action are visible in the second row of Table 8.3. Current state #6 in combination with look-ahead item *student* elicits a reduction step ("Rd Art"). This causes *the* to be replaced by Art, corresponding to rule 15 of the sample grammar. At the same time, the processor enters state #0 again.

[iv] Algorithms for converting a grammar to a control table are discussed by Aho and Ullman (1972).

[v] The numbers are assigned by the algorithm that constructs the control table. There is no relationship between these numbers and those on the ATN arcs in Figure 8.2.

Table 8.2. Control table corresponding to the sample grammar in Table 8.1.

	Art	N	NP	PP	Prep	PropN	RRC	S	VP	Vintr	Vpass	Vtr	#	Chrys	fainted	letter	read	student	the	to
0	1		2			3		4						Sh, 5					Sh, 6	
1		7														Sh, 8		Sh, 9		
2				10	11		12		13	14	15	16			Sh, 17		Sh, 18			Sh, 19
3													NP		NP		NP			NP
4													†							
5													PropN		PropN		PropN			PropN
6																Art		Art		
7													NP		NP		NP			NP
8													N		N		N			N
9													N		N		N			N
10													NP		NP		NP			NP
11	1		20			3								Sh, 5					Sh, 6	
12													NP		NP		NP			NP
13													S							
14				21	11								VP							Sh, 19
15	1		22			3								Sh, 5					Sh, 6	
16	1		23			3								Sh, 5					Sh, 6	
17													Vintr							Vintr
18														Vtr Vpass					Vtr Vpass	
19														Prep					Prep	
20				10	11		12				15		PP		PP		PP Sh, 24			PP Sh, 19
21													VP							
22				10	11		12				15		RRC		RRC		RRC Sh, 24			RRC Sh, 19
23				25	11		12				15		VP				Sh, 24			Sh, 19
24														Vpass					Vpass	
25													VP NP		NP		NP			NP

Table 8.3. Shift-Reduce Parsing of example (16b).

Stack	Input Buffer	Action
Ø-0	the student read ...	Shift, →6
Ø-0 the-6	student read the ...	Rd Art
Ø-0 Art	student read the ...	→1
Ø-0 Art-1	student read the ...	Shift, →9
Ø-0 Art-1 student-9	read the letter ...	Rd N
Ø-0 Art-1 N	read the letter ...	→7
Ø-0 Art-1 N-7	read the letter ...	Rd NP
Ø-0 NP	read the letter ...	→2
Ø-0 NP-2	read the letter ...	Shift, →18
Ø-0 NP-2 read-18	the letter to ...	*RdVtr*/Rd Vpass
Ø-0 NP-2 Vtr	the letter to ...	→16
Ø-0 NP-2 Vtr-16	the letter to ...	Shift, →6
Ø-0 NP-2 Vtr-16 the-6	letter to Chrysanne ...	Rd Art
Ø-0 NP-2 Vtr-16 Art	letter to Chrysanne ...	→1
Ø-0 NP-2 Vtr-16 Art-1	letter to Chrysanne ...	Shift, →8
Ø-0 NP-2 Vtr-16 Art-1 letter-8	to Chrysanne fainted #	Rd N
Ø-0 NP-2 Vtr-16 Art-1 N	to Chrysanne fainted #	→7
Ø-0 NP-2 Vtr-16 Art-1 N-7	to Chrysanne fainted #	Rd NP
Ø-0 NP-2 Vtr-16 NP	to Chrysanne fainted #	→23
Ø-0 NP-2 Vtr-16 NP-23	to Chrysanne fainted #	Shift, →19
Ø-0 NP-2 Vtr-16 NP-23 to-19	Chrysanne fainted #	Rd Prep
Ø-0 NP-2 Vtr-16 NP-23 Prep	Chrysanne fainted #	→11
Ø-0 NP-2 Vtr-16 NP-23 Prep-11	Chrysanne fainted #	Shift, →5
Ø-0 NP-2 Vtr-16 NP-23 Prep-11 Chrysanne-5	fainted #	Rd PropN
Ø-0 NP-2 Vtr-16 NP-23 Prep-11 PropN	fainted #	→3
Ø-0 NP-2 Vtr-16 NP-23 Prep-11 PropN-3	fainted #	Rd NP
Ø-0 NP-2 Vtr-16 NP-23 Prep-11 NP	fainted #	→20
Ø-0 NP-2 Vtr-16 NP-23 Prep-11 NP-20	fainted #	Rd PP
Ø-0 NP-2 Vtr-16 NP-23 PP	fainted #	→25
Ø-0 NP-2 Vtr-16 NP-23 PP-25	fainted #	Rd NP
Ø-0 NP-2 Vtr-16 NP	fainted #	→23
Ø-0 NP-2 Vtr-16 NP-23	fainted #	Fail

The remainder of this Section will deal with ambiguity, following some of the ideas expressed by Pereira (1985). The action in the tenth row shows what happens when lexical ambiguity cannot be resolved by look-ahead. (Other cases of lexical ambiguity *are* solvable by look-ahead. For instance, *that* followed by *books* will be a subordinating conjunction rather than a demonstrative pronoun; e.g. *You know that books can be heavy*.) The word *read* is ambiguous between transitive (active) and passive verb (Vtr or Vpass), and the control table offers two possible reductions. In order to solve this reduce-reduce conflict the processor has recourse to external information, e.g., the relative frequencies of *read*'s lexical frames. I have assumed that *read* in this particular passive construction (exemplified by *the student was read the letter* rather than by *the letter was read to the student*) is less common than *read* as transitive active verb. This selects the reduction corresponding to grammar rule *Vtr → read*. This choice will turn out to have fatal consequences. The control table entry corresponding to bottom row configuration 'state #23, *fainted*' is empty, indicating parser failure. In other words, the processor has been led up the garden path.

Sentence (16a) gives rise to a happier course of events. Table 8.4 shows the parsing steps after *Chrysanne* has been shifted onto the stack. The seventh row reveals a reduce-reduce conflict, namely, between rule 10 (*VP → Vtr NP PP*) and rule 4 (*NP → NP PP*) of the sample grammar. The former replaces three stack symbols, the latter only two. Pereira recommends to select the "longer" option if one desires Minimal Attachment. The former reduction indeed corresponds to Figure 8.4a.

Pereira's second advice concerns shift-reduce conflicts. Consider the course of events when *read* is analyzed as a Vpass and *the letter* as an NP (Table 8.5). The control table offers a choice between shifting look-ahead item *to* onto the stack, or a reduction according to grammar rule *RRC → Vpass NP* (see last line of Table 8.5). The shift option has the advantage of yielding Right Association: the PP *to Chrysanne* will become attached as the sister of *the letter* rather than as a higher node. Pereira has shown that solving shift-reduce conflicts in favor of shifting guarantees Right Association (see also Shieber, 1983).

Table 8.4. Shift-Reduce Parsing of example (16a): final part.

Stack	Input Buffer	Action
Ø-0 NP-2 Vtr-16 NP-23 Prep-11 Chrysanne-5	#	Rd Prep
Ø-0 NP-2 Vtr-16 NP-23 Prep-11 PropN	#	→ 3
Ø-0 NP-2 Vtr-16 NP-23 Prep-11 PropN-3	#	Rd NP
Ø-0 NP-2 Vtr-16 NP-23 Prep-11 NP	#	→20
Ø-0 NP-2 Vtr-16 NP-23 Prep-11 NP-20	#	Rd PP
Ø-0 NP-2 Vtr-16 NP-23 PP	#	→ 25
Ø-0 NP-2 Vtr-16 NP-23 PP-25	#	*Rd VP*/Rd NP
Ø-0 NP-2 VP	#	→13
Ø-0 NP-2 VP-13	#	Rd S
Ø-0 S	#	→4
Ø-0 S-4	#	†

Control Table 8.2 contain cells that leave a choice between two or more possible actions. If the LR parser indeed explores several or all of these options in parallel or sequentially, the parser is non-deterministic. In contrast, if an LR control table does not contain a cell that specifies more than one possible action, the parser is said to be *deterministic*. Adding *conflict resolution strategies* changes a non-deterministic parser into a deterministic one. From a psycholinguistic point of view this is an attractive move because, as we have seen in Section 8.2.1, the human syntactic processor avoids parallel ("multiple-track") processing. Garden-path phenomena, however, imply that our parsing mechanism is not fully deterministic for they induce backtracking, that is, sequential exploration of alternative syntactic options. I will return to this issue in the next Section.

A psychological phenomenon that seems to pose a problem for Shift-Reduce parsers is *incremental* analysis. Take rule 10 of the sample grammar in Table 8.1: *VP → Vtr NP PP*. The corresponding reduction can be executed only after the parser has identified all three VP subconstituents. Up to that point, the constituents are just waiting on the stack while their role in the sentence structure is left undecided. An incremental parser, on the contrary, will attempt to assign every subconstituent, if not every single word, a grammatical function without delay. Abney (1989) has proposed an extension of LR(1) parsing, called Licensing-Structure parsing, intended to remedy this problem. However, this model has recently come under attack from an empirical angle (Hemforth, Konieczny, & Strube, 1993). A theoretical solution to the incremental analysis problem for LR parsers has been proposed by Shieber and Johnson (1993).

Table 8.5. Shift-Reduce Parsing of example (16b): correct analysis replacing the last nine steps shown in Table 8.3.

Stack	**Input Buffer**	**Action**
Ø-0 NP-2 read-18	the letter to ...	Rd Vtr/*Rd Vpass*
Ø-0 NP-2 Vpass	the letter to ...	→ 15
Ø-0 NP-2 Vpass-15	the letter to ...	Shift, → 6
Ø-0 NP-2 Vpass-15 the-6	letter to Chrysanne ...	Rd Art
Ø-0 NP-2 Vpass-15 Art	letter to Chrysanne ...	→ 1
Ø-0 NP-2 Vpass-15 Art-1	letter to Chrysanne ...	Shift, →8
Ø-0 NP-2 Vpass-15 Art-1 letter-8	to Chrysanne fainted ...	Rd N
Ø-0 NP-2 Vpass-15 Art-1 N	to Chrysanne fainted ...	→ 7
Ø-0 NP-2 Vpass-15 Art-1 N-7	to Chrysanne fainted ...	Rd NP
Ø-0 NP-2 Vpass-15 NP	to Chrysanne fainted ...	→ 22
Ø-0 NP-2 Vpass-15 NP-22	to Chrysanne fainted ...	Rd RRC/*Shift, → 19*

8.3.4 PARSIFAL

Although natural language is fraught with ambiguity, and this chapter with talk about garden-paths, fact is that people seldom become consciously aware of having misparsed a sentence. PARSIFAL is an attempt at designing a single-track parser which fails at exactly those sentential positions where people become consciously aware of having been misled, and only then undertakes to reanalyze the sentence (backtracking). If a parse fails (presumably due to a garden-path), an external reanalysis/recovery mechanism is activated which diagnoses the situation and puts the parser back on track. In other words, PARSIFAL's design aimed at exactly mirroring the degree of determinism[vi] of the human syntactic parser.

Marcus described his work on PARSIFAL in his 1977 Ph.D. dissertation, which was published in 1980. Here I can only render the bare essentials of the parser, ignoring the reanalysis component which helps it to recover from garden-paths. PARSIFAL employs a stack and

[vi] If the reanalysis/recovery component is not considered to belong to the parsing mechanism proper, PARSIFAL may be said to be fully deterministic.

condition-action rules not unlike normal Shift-Reduce parsers[vii]. Its most important distinguishing feature concerns the treatment of look-ahead, which is not restricted to the first undigested input word. The input buffer is conceived of as a row of cells, and the three left-most cells are accessible for look-ahead. Initially, each cell is occupied by a single word. However, the buffer is allowed to include non-terminal symbols from the grammar, e.g. NP or VP. This is needed, for instance, in cases such as (17),

(17) a. Have the new students taken the exam today?

b. Have the new students take the exam today.

where the processor has no way of telling whether *have* is an auxiliary or a main verb without first inspecting the verb following *the new students*. If this second verb is an infinitive, *have* is a main verb; is it a past participle, then *have* is auxiliary. Since most native speakers of English appear to parse these sentences without being garden-pathed, the look-ahead must include *take(n)*. But the NP in-between *have* and *take(n)* is already three words long, so *take(n)* is inaccessible to the processor. Expanding the look-ahead to four or even more words does not solve the problem because the intervening NP can be arbitrarily long. Marcus' solution, in principle, amounts to activating a second instantiation of the parser, assigning it to find an NP in the remaining string, and inserting the complete NP into the first buffer cell (cf. Berwick & Weinberg, 1984, p. 280). After this intermediate operation (called "attention shifting" by Marcus), the first look-ahead item comprises the complete NP rather than its leading edge *the*; and *take(n)*, promoted to the second buffer cell, is within the processor's scope.

Although the model has attracted a great deal of attention, from an empirical point of view it was not very successful. Pritchett (1992, p. 44ff) lists some of the mismatches between predicted and observed garden-paths. Sentence (18) is short enough to fit into the input buffer but causes problems nevertheless. Similarly, in (19) the distance between *her* and *would* is small enough for the processor to decide that *her* is a personal rather than a possessive pronoun.

(18) Boys hit cry. (Boys who are hit cry.)

(19) Without her money would be hard to come by.

At the end of his book, Marcus himself admits that the size of the look-ahead buffer may vary among individuals, and a few years later the model is substantially revised so as to improve the empirical coverage (see Marcus, Hindle & Fleck's, 1983). Pritchett (1992), however, points out that problems remain. Finally, as to the nature of syntactic-semantic integration, Marcus argues that deterministic parsing not only allows but even necessitates an interactive model. The condition-action rules must be sensitive to semantic and pragmatic factors if premature structure building is to be avoided. This aspect has not been implemented, however.

[vii] Although PARSIFAL is an independent development, Berwick and Weinberg (1984, p. 185ff.) have shown that it may be viewed as a special kind of Shift-Reduce parser. I will follow their lead.

8.3.5 Sausage Machine and Race-Based Parsing

In the late seventies, Frazier and Fodor developed a well-known parsing model called the Sausage Machine (see Frazier & Fodor, 1978 and Fodor & Frazier, 1980). Taking this model as their point of departure, McRoy and Hirst (1990) developed a computational sentence processor called Race-Based Parsing that accounts for a considerably wider range of empirical phenomena than any of the models discussed so far. Before going into Race-Based Parsing, I will first outline the essence of the Sausage Machine.

Sausage Machine

The Sausage Machine consists of two cascaded parsing stages during which the same grammar is used. In Stage 1, input words are parsed, resulting in relatively simple phrases or clauses. These chunks are shunted to Stage 2 which combines them to complete sentences. Although their inputs are very different, the stages operate similarly. Their working memories are both limited to about six units (words and phrases/clauses respectively), and they abide by the same, now familiar parsing principles of Minimal Attachment and Right Association. There is no feedback from the second to the first parsing stage.

Minimal Attachment (see Section 8.3.2) is invoked to account not only for various garden-paths such as (3a) and (16b) but also for the difficulty of understanding multiply center embedded clauses. The difficulty of (20) is supposed to stem from a tendency to interpret the initial NPs as a coordination (cf. *The woman, the man, the girl and ...*).

(20) The woman the man the girl loved met died.

An attractive feature of the Sausage Machine is its capability to explain certain interactions between the parsing principles. For instance, in the preferred interpretation of (21a), the final PP is attached to the VP rather than to the NP. This preference is reversed in (21b).

(21) a. John read the letter to Mary.

b. John read the note, the memo and the letter to Mary.

The reason is that the words of (21a) fit into the working memory of Stage 1, which therefore can parse the sentence as a whole while satisfying the principle of Minimal Attachment. But (21b) is too long. The first stage only sees *John read the note, the memo*, parses this as a clause, and shunts it to Stage 2. Spotting then *and the letter to Mary*, Stage 1 can only parse this fragment as a single NP because the verb *read* is out of sight. Stage 2 accepts this input and appends it unchanged to the clause received earlier. Notice that this course of events presupposes the assumption of determinism.

Frazier and Fodor provide few details concerning the style of parsing employed by their Sausage Machine. They suggest that the attachment of an incoming lexical item depends on the outcome of a *race*. In particular, the processor explores in parallel various ways of relating the new item to the already existing structure, and the first alternative meeting with success is favored. A mechanism of this sort could be responsible for the Minimal Attachment preferences. Non-minimal attachment requires accessing more grammar rules, which presumably takes more time.

The assumption of parallel computing involved here does not contradict the notion of bounded parallelism that is advocated in Section 8.2.1 as a plausible form of process control in the human syntactic processor. This is because after the winner of a race has been selected, all its competitors are thrown away. It may be useful to introduce a distinction between *local* and *global* parallelism. The latter involves parallel exploration of alternative attachments of individual input items without choosing a winner and destroying the losers. That is, some or all of the alternative analyses are kept until the end of the sentence or a disambiguation point has been reached. Local parallelism means that one of the explored attachment alternatives is selected as best, at least provisionally, and all traces of the competitors are removed.

Race-Based Parsing

McRoy and Hirst adopted the basic idea of processing races in a modified form. For each of the attachment alternatives they calculate a *time cost*, and the winning (cheapest, fastest) one will the guide further processing. Furthermore, time cost not only depends on number of grammar rules involved in an attachment. Other cost determining factors are Priming, Distance (Right Association or Recency; cf. Section 8.2.4), Lexical Frame Preferences (cf. Section 8.2.2) and Semantic Preferences. The resulting architecture is interactive in the sense of Section 8.2.1.

The central component of the Race-Based Parser is the *Attachment Processor*. Consulted by Stage One or Stage Two to suggest a suitable attachment point for a new item (a word or a phrase/clause), it calls on syntactic, semantic/pragmatic and lexical procedures (so-called hypothesizers) to suggest possible alternatives and their associated time costs, and commands tree formation routines to actually carry out the lowest-cost attachment.

McRoy and Hirst illustrate their model by tracing through the processing of sentence (16a). From their account I select the steps which are of greatest interest in the present context. The sentence is short enough to be fully processed by Stage 1. When *read* is processed, the lexicon suggests both the Vtr and the Vpass options. One of the hypothesizers reports that the latter option will be more costly because building a reduced relative clause takes more time than a finite clause (six versus three cost units; the authors admit that the numbers are somewhat arbitrary). Attaching *read* as a transitive verb leads the parser to predict an object NP. This expectation is fulfilled by the next word. The article *the* is attached to the tree, thereby creating the expectation of a noun. After *letter* has been attached accordingly, the highly ambiguous *to* is considered next. The option of *to* as a complementizer (e.g. … *to pass the time*) is very expensive (eleven units). For the preposition *to*, the hypothesizers offer various possibilities: as a modifier to *letter* (four units), as a modifier to *read* (eleven units; the distance to *read* is larger than to *letter*), and as the indirect object. The latter option has a time cost of only three units because it fulfils *read*'s semantic expectation of a Beneficiary argument. The end result is a parse which conforms to the preferred interpretation of (16a). Incidentally, notice that the choice in favor of the VP rather than the NP attachment was not dictated by Minimal Attachment in the original sense but by expectations raised by lexical frame information.

I cannot go into further details of the time cost calculations or the actual implementation. The outline presented here suffice to show that Race-Based Parsing, given appropriate time cost functions, in principle can account for psycholinguistic phenomena related to syntactic recency, semantic-syntactic interaction, priming, and lexical frame preferences. The two latter effects are simulated by allotting fewer time cost units to, respectively, recently processed constructions and preferred lexical frames. Moreover, unlike any of the previous models, the implementation includes a rudimentary revision component which allows recovery from mild parsing failures.

However, the model's basic architecture is at variance with MacDonald *et al.*'s (1992) finding concerning lexical frame ambiguity, discussed in Section 8.2.1 in the context of examples (3) and (4). In terms of example (16a), input item *read* will lead to the selection of either the Vtr or the Vpass option; the fact that this word has two entries in the lexicon rather than one, does not slow down the parallel exploration of attachment alternatives. Nor does it affect the complexity of subsequent processing because the traces of the losing option are immediately erased. Saving losing options would go against the grain of the overall architecture and its source of inspiration, the Sausage Machine.

8.3.6 The Unification Space

The Unification Space model, proposed by Kempen and Vosse (1989; see also Vosse & Kempen, 1991), proceeds from a chemical synthesis metaphor. The molecules entering into bonds are lexical trees with nodes as potential attachment points. For every word of the language the lexicon specifies at least one lexical tree, each having one or more attachment points (see Figure 8.6).

Two trees dominating different words of a sentence may combine into a larger tree by merging attachment points, a process called unification (Figure 8.7). The selection proceeds on a probabilistic basis. The unification probabilities p(U) between two nodes depend on various lexical, syntactic and semantic/pragmatic factors: certain node pairs make better unification partners than others. The determinants of "goodness of fit" are summarized in one variable called *Strength*. Probabilities p(U) correlate positively with Strength.

In contrast with the foregoing models, attachments may break up, that is, unifications may be canceled so that the nodes become available for other, possibly stronger unifications. Break-up probabilities p(B) correlate negatively with Strength. It follows that candidates forming a good partnership are more likely to unify and less likely to break up: the stronger a unification, the longer it will survive. The consequence is that, as time proceeds, more and more nodes will find a strong and stable unification partner, and more and more lexical trees will cluster together. As soon as this search/optimization process has settled down, the sentence has been parsed. If the result is a tree that spans the whole sentence, the parse is successful. In fact, the model delivers at most one such tree for a grammatical sentence, in accordance with the characteristic of single-track processing (see Section 8.2.1). Parsing has failed if the result consists of several disconnected trees, each dominating only part of the sentence.

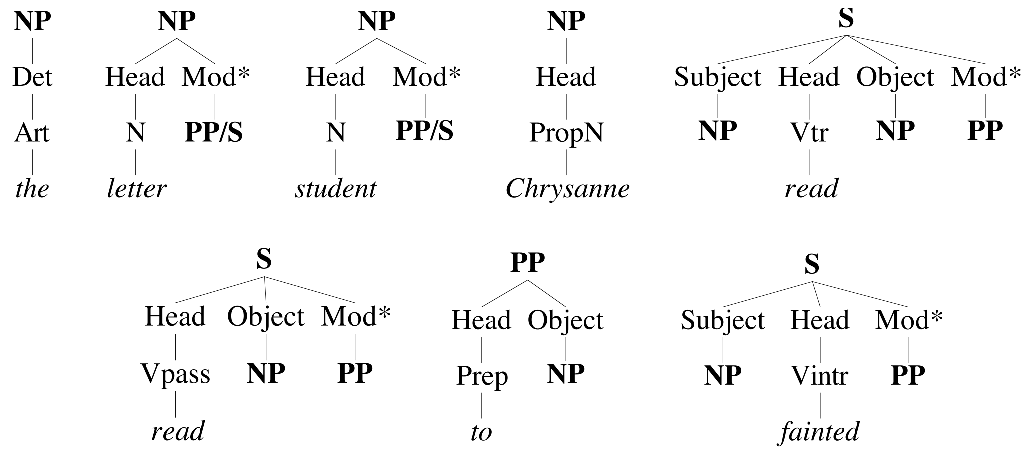


Figure 8.6. Lexical trees corresponding to the words of examples (16a/b). Attachment points printed in bold.

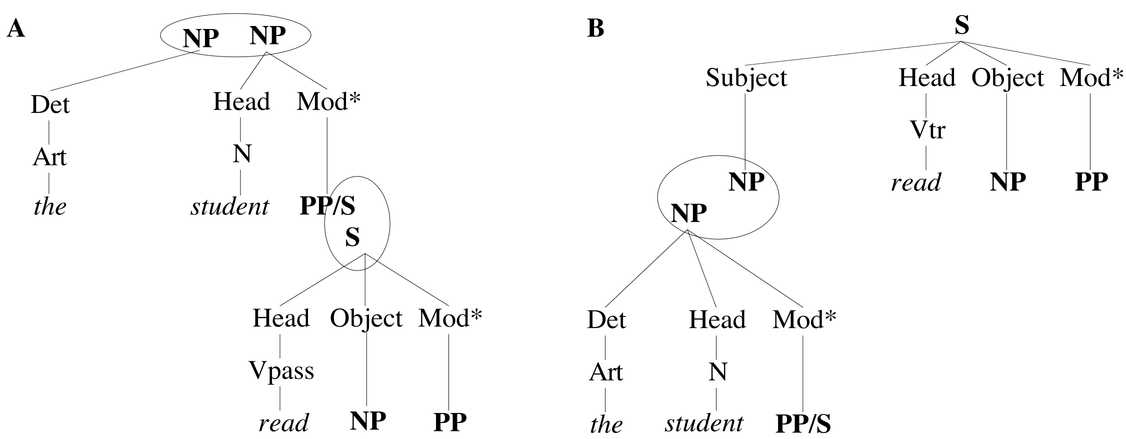


Figure 8.7. Two possible unifications of some of the lexical frames of Figure 8.6.

As shown in Figure 8.5, there is a tree for every lexical frame associated with a word. The root and the non-lexical terminals of a tree are attachment points. An asterisk on a Modifier node indicates that this node and the branch it belongs to may occur zero or more times. Only identically labeled nodes are allowed to unify, e.g. the roots of the trees for *the* and *student*, or the S nodes that dominate Vpass and modify *student*, respectively. As to the concept of unification, Kempen and Vosse use a non-recursive form of feature unification; one of its duties is to check agreement (number, person, gender, etc.) between unification partners.

It is important to note that all rules of syntax are encoded in the lexical trees and the unification operation. The "Segment Grammar" formalism adopted in the Unification Space model (for some details see Chapters 2 and 11) therefore belongs to the class of "lexicalized grammars" together with, e.g., Tree Adjoining Grammar (cf. Rambow & Joshi 1994).

While input words are read from the buffer one by one, the lexical tree(s) associated with each of them enter(s) the "Unification Space" (U-Space for short). The "free" nodes (attachment points) start hunting for suitable unification partners without delay. However, due to the probabilities p(U) and p(B), the attachments may never be considered final. The possibility of a break-up always lurks as long as the process has not come to a halt.

The dynamics of the model depend to a large extent on the level of *activation* of syntactic nodes[viii] Upon entry into U-Space, every node of a lexical tree is assigned an activation level. The entry levels correlate positively with frequency in the language and with the language user's preference (if that is something different). Activation is supposed to decrease by a constant fraction per time unit ("decay"). Activation has a strong positive influence on probabilities p(U). This causes the more active nodes to be the more likely unification partners. In other words, active nodes are more avid explorers of the search space. However, due to the unremitting decay of activation levels, the avidity of all nodes is bound to decrease and the system will come to rest.

Empirical predictions are derivable from average parsing times and proportions of successful parses when a sentence is presented to the model many times (Monte Carlo simulation method). Input sentences that consume more processing cycles or give rise to more parsing failures, are predicted to be harder to understand for human language users. The model's interpretive biases reveal themselves in the alternative parse trees delivered during a Monte Carlo run: preferred analyses will turn up in greater proportions. Kempen and Vosse report successful simulations of an interesting range of psycholinguistic phenomena. The Syntactic Recency effect follows from the higher levels of activation of more recently launched lexical trees. Lexical Frame Preferences and Syntactic Priming are modeled in terms of varying entry level activations. The garden-path character of example (16b) results from the low frequency of *read* occurring in that particular lexical frame. The comprehensibility ranking of three types of embedded clauses (cf. examples (15a,b,c) in Section 8.2.4) was "pre"dicted correctly. The U-Space belongs to the class of interactive models due to the semantic/pragmatic contribution to the Strength parameter.

Notwithstanding these successes, various problems remain to be solved. For instance, constituent length effects (cf. Section 8.2.3) do not fit into the 1989 version of the model nor do the syntactic frequency phenomena discussed in Section 8.2.2.

8.4 CONCLUSIONS AND OUTLOOK

A fair conclusion from the preceding survey of syntactic processing models is that the gulf between requirements (Section 8.2) and achievements (Section 8.3), although slowly and steadily narrowing down, is still looming large. Neither the experimentalists nor the modelers are to be blamed for this. The models were under construction *while* the empirical domain was being opened up through increasingly sophisticated machinery so that, of necessity, they have been aiming at moving targets.

[viii] For a characterization of activation-based models, see Chapter 3.

Recent developments are the discovery that syntactic processing problems and preferences are sensitive to statistical factors (addressed in Section 8.2.3), and the spread of dynamic modeling techniques in the wake of connectionist successes. Since statistical trends are readily expressible in dynamic terms, one may expect a keen interest of experimentalists in dynamic modeling of human syntactic processing. Another reason why dynamic models may hold great attraction for experimental psycholinguists is the *graded* nature of their empirical predictions, as opposed to merely *all-or-none* (binary) predictions allowed by most structural models. Examples of dynamic models are not only Race-based Parsing and the Unification Space but also more recent neural network models (e.g. Stevenson, 1993a; 1993b and Henderson, 1992).

Dynamic/stochastic models are not without danger, though. Because they tend to contain quite a few numerical parameters, they run the risk of losing predictive power in as far as their behavior is potentially tweaked to any desired pattern. They may also fail to meet linguistic demands with respect to descriptive or explanatory power. To my judgment, none of these dangers is threatening the dynamic models described or quoted in this chapter; they are firmly rooted in modern syntactic theory.

What will be the main concerns of computational modelers in the years to come? Central topics will undoubtedly include the design of computational architectures that account for semantic-syntactic interactions and for statistical effects in empirically justifiable ways. The behavior of these systems should be tested in the context of considerably larger grammars and lexica than the toy versions that current models have to make do with. Another topic—of utmost importance but hardly explored—is the relation between syntactic processes in sentence comprehension and those in sentence *production*. Finally, it is my personal conviction that progress in these areas presupposes a multidisciplinary research setting with combined experimental-psychological, computational and linguistic expertise.

Acknowledgements
I am indebted to Theo Vosse for numerous discussions about all aspects of parsing. His parser software generated the LR(1) control table (Table II). I thank Don Mitchell, the editors, and three anonymous reviewers for their comments on an earlier version of this chapter.

8.5 References

Abney, S.P. (1989). A computational model of human parsing. *Journal of Psycholinguistic Research*, *18*, 129-144.

Aho, A. & J. Ullman, (1972). *The theory of parsing, translation and compiling*. Englewood Cliffs NJ: Prentice-Hall.

Altmann, G.T.M., A. Garnham & Y. Dennis (1992). Avoiding the garden-path: eye movements in context. *Journal of memory and language*, *31*, 685-712.

Bach, E., C. Brown & W. Marslen-Wilson (1986). Crossed and nested dependencies in German and Dutch: a psycholinguistic study. *Language and cognitive processes*, *4*, 249-262.

Berwick, R.C. & A.S. Weinberg (1984). *The grammatical basis of linguistic performance*. Cambridge MA: MIT Press.

Britt, M.A. (1994). The interaction of referential ambiguity and argument structure in the parsing of prepositional phrases. *Journal of Memory and Language*, *33*, 251-283.

Clifton, C., L. Frazier & C. Connine (1984). Lexical expectations in sentence comprehension. *Journal of verbal learning and verbal behavior*, *23*, 696-708.

Crain, S. & M. Steedman (1985). On not being led up the garden path: the use of context by the psychological syntax processor. In: D. Dowty, L. Karttunen & A. Zwicky (Eds.) *Natural language parsing: Psychological, computational, and theoretical perspectives*. Cambridge UK: Cambridge University Press.

Cuetos, F. & D.C. Mitchell (1988). Cross-linguistic differences in parsing: restrictions on the use of the Late Closure strategy in Spanish. *Cognition*, *30*, 73-105.

Cuetos, F., D.C. Mitchell & M.M.B. Corley (in press). Parsing in different languages. In: M. Carreiras, J. Garcia-Albea & N. Sebastian-Galles (Eds.) *Language processing in Spanish*. Hillsdale NJ: Erlbaum.

Fodor, J.D. (1989). Empty categories in sentence processing. *Language and cognitive processes*, *4*, 155-209.

Fodor, J.D. & L. Frazier (1980). Is the human sentence parsing mechanism an ATN? *Cognition*, *8*, 417-459.

Frazier, L. (1978). *On comprehending sentences: syntactic parsing strategies*. Ph.D. Thesis, University of Connecticut.

Frazier, L. & G.B. Flores d'Arcais (1989. Filler driven parsing: a study of gap filling in Dutch. *Journal of memory and language*, *28*, 331-344.

Frazier, L. & J.D. Fodor (1978). The sausage machine: a new two-stage parsing model. *Cognition*, *6*, 291-325.

Frazier, L. & K. Rayner (1982). Making and correcting errors during sentence comprehension: eye movements in the analysis of structurally ambiguous sentences. *Cognitive Psychology*, *14*, 178-210.

Frazier, L., L. Taft, T. Roeper, C. Clifton & K. Ehrlich (1984). Parallel structure: a source of facilitation in sentence comprehension. *Memory and cognition*, *12*, 421-430.

Garrett, M.F. (1990). Sentence processing. In: D.N. Osherson & H. Lasnik (Eds.) *An invitation to cognitive science*. Cambridge MA: MIT Press.

Gibson, E. (1990). Recency preference and garden-path effects. *Proceedings of the 12th Cognitive Science Society*.

Gibson, E. & J. Loomis (1994). A corpus analysis of recency preference and predicate proximity. *Proceedings of the 16th Annual Conference of the Cognitive Science Society*. Hillsdale NJ: Erlbaum.

Gorrell, P. (1991). Subcategorization and sentence processing. In: R.C. Berwick, S.P. Abney & C. Tenny (Eds.) *Principle-based parsing: computation and psycholinguistics*. Dordrecht/Boston: Kluwer Academic Press.

Hemforth, B., L. Konieczny & G. Strube 1993). Incremental syntax processing and parsing strategies. *Proceedings of the 15th Annual Conference of the Cognitive Science Society*. Hillsdale NJ: Erlbaum.

Henderson, J. (1992). A connectionist parser for structure unification grammar. *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*.

Kaplan, R.M. (1972). Augmented transition networks as psychological models of sentence comprehension. *Artificial Intelligence*, *3*, 77-100.

Kaplan, R.M. (1975). On process models for sentence analysis. In: D.A. Norman, D.E. Rumelhart & the LNR Research Group (Eds.) *Explorations in cognition*. San Francisco: Freeman.

Kempen, G. & Th. Vosse (1989). Incremental syntactic tree formation in human sentence processing: a cognitive architecture based on activation decay and simulated annealing. *Connection Science, 1*, 273-290.

Kimball, J. (1973). Seven principles of surface structure parsing in natural language. *Cognition*, *2*, 15-47.

MacDonald, M.C., M.A. Just & P.A. Carpenter (1992). Working memory constraints on the processing of syntactic ambiguity. *Cognitive Psychology*, *24*, 56-98.

Marcus, M.P. (1980). *A theory of syntactic recognition for natural language*. Cambridge MA: MIT Press.

Marcus, M.P., D. Hindle & M.M. Fleck (1983). D-Theory: Talking about talking about trees. *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*.

McRoy, S.W. & G. Hirst (1990). Race-Based Parsing and syntactic disambiguation. *Cognitive Science*, *14*, 313-353.

Mitchell, D.C. (1994). Sentence parsing. In: M.A. Gernsbacher (Ed.) *Handbook of psycholinguistics*. San Diego: Academic Press.

Mitchell, D.C. (1987). Lexical guidance in human parsing: locus and processing characteristics. In: M. Coltheart (Ed.), *Attention and performance XII: the psychology of reading*. Hillsdale NJ: Erlbaum.

Pereira, F.C.N. (1985). A new characterization of attachment processes. In: D.R. Dowty, L. Karttunen & A.M. Zwicky (Eds.) *Natural language parsing: Psychological, computational, and theoretical perspectives*. Cambridge UK: Cambridge University Press.

Pickering, M. & G. Barry (1991). Sentence processing without empty categories. *Language and cognitive processes, 6*, 229-259.

Pritchett, B.L. (1992). *Grammatical competence and parsing performance*. Chicago: The University of Chicago Press.

Rambow, O. & Joshi, A. (1994). A processing model for free word-order languages. In: C. Clifton, L. Frazier & K. Rayner (Eds.), *Perspectives on sentence processing* (pp. 267-301). Hillsdale, NJ: Erlbaum.

Rayner, K., S. Garrod & C.A. Perfetti (1992). Discourse influences during parsing are delayed. *Cognition*, *45*, 109-139.

Rayner, K., J.M. Pacht & S.A. Duffy (1994). Effects of prior encounter and global discourse bias on the processing of lexically ambiguous words: evidence from eye fixations. *Journal of Memory and Language*, *33*, 527-544.

Schuster, S. & M.C. MacDonald (1994). Syntactic priming and syntactic ambiguity resolution. Poster presented at the Seventh CUNY Conference on Human Sentence Processing.

Shapiro, L.S., H.N. Nagel & B.A. Levine (1993). Preferences for a verb's complements and their use in sentence processing. *Journal of memory and language*, *32*, 96-114.

Shieber, S.M. (1983). Sentence disambiguation by a Shift-Reduce parsing technique. *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics.*

Shieber, S. & M. Johnson (1993). Variations on incremental interpretation. *Journal of Psycholinguistic Research*, *22*, 287-318.

Stevenson, S. (1993a). A competition-based explanation of syntactic attachment preferences and garden path phenomena. *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics.*

Stevenson, S. (1993b). Establishing long-distance dependencies in a hybrid network model of human parsing. *Proceedings of the 15th Annual Conference of the Cognitive Science Society.* Hillsdale NJ: Erlbaum.

Swinney, D.A. (1979). Lexical access during sentence comprehension: (Re)consideration of context effects. *Journal of verbal Learning and verbal behavior*, *18*, 654-660.

Taraban, R. & J.L. McClelland (1988). Constituent attachment and thematic role assignment in sentence processing: influences of content-based expectations. *Journal of memory and language*, *27*, 597-632.

Trueswell, J.C., M.K. Tanenhaus & S.M. Garnsey, (1994). Semantic influences on parsing: use of thematic role information in syntactic ambiguity resolution. *Journal of Memory and Language*, *33*, 285-318.

Vosse, Th. & G. Kempen (1991). A hybrid model of human sentence processing: parsing right-branching, center-embedded and cross-serial dependencies. In: *Proceedings of the Second International Workshop on Parsing technologies.*

Wanner, E. (1980). The ATN and the Sausage Machine: Which one is baloney? *Cognition*, *8*, 209-225.

Winograd, T. (1983). *Language as a cognitive process. Volume I: Syntax.* Reading MA: Addison-Wesley.