North East Linguistics Society

Volume 15
Issue 1 *NELS 15*

Article 18

1985

A Model of Syntactic Phrase Combination During Speech Production

Steven G. Lapointe
Wayne State University

A Model of Syntactic Phrase Combination During Speech Production

Steven G. Lapointe

Wayne State University

In a series of recent articles (Lapointe 1983, 1984, to appear a,b), I have attempted to formulate a theory of the storage and access of information about grammatical markers during the syntactic processing phase of speech production. However, in those works I said remarkably little about the ways in which this information was subsequently combined into larger syntactic units, a situation I would like to begin to rectify in the present paper. The research I will be reporting on is part of a larger work currently in progress which seeks to incorporate a number of results about syntactic and lexical processing, as well as several important psychological notions concerning attention and types of processing operations, into a broader theoretical organization. My remarks will fall into three main parts. First, I will briefly outline the system of grammatical marker storage and access from my earlier papers. Next, I will sketch a model of syntactic phrase combination which forms the backbone of the syntactic production processor. Finally, I will consider the syntactic speech error data to test some of the preliminary consequences of the model.

Before I launch into the discussion, however, some caveats are in order. I will not be considering the speech-timing results of Cooper and Paccia-Cooper (1980) since these results involve issues concerning the syntax/phonology interface that are considerably less clear than I at first thought. I also will not be saying anything about lexical influences on the selection of syntactic structures of the sort discussed, for example, by Bock (1982). For present purposes, we can get pretty far without worrying about this extra influence on syntactic processing. In a more comprehensive study, we would of course want to include a discussion of such influences. Finally, the model of grammatical marker processing that I have alluded to has undergone several changes over the past year; where there are differences between earlier and later versions of the model, I will be relying on the proposals in the most recent article, an as yet unpublished work, Lapointe (1984). With these preliminaries out of the way, let us now move on to grammatical marker processing.

1. Earlier Proposals about Syntactic Processing

The model adopted in Lapointe (1984) is based on a model proposed in earlier work on normal speech errors by Garrett (1975). The overall organization of that model is given in Figure 1.

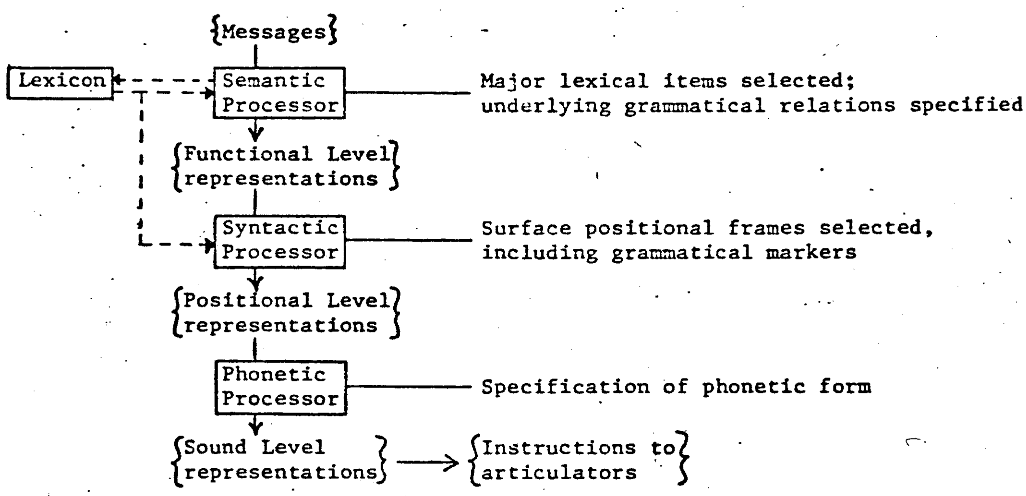


Figure 1. General organization of production processes, after Garrett (1975).

In this system, an utterance is assumed to be produced in the following way. At some point, the speaker decides to express some set of Internal Messages verbally. These Messages are sent to the Semantic Processor which selects major lexical items from the Mental Lexicon and specifies the underlying "grammatical relations" of the utterance. In doing this, the Semantic Processor creates what Garrett calls the Functional Level representation. This representation is then input to the Syntactic Processor which selects "positional frames", that already contain grammatical markers in their surface syntactic positions, and inserts the major lexical items previously selected into the appropriate slots in these frames. These operations result in Positional Level representations which are then input to the Phonetic Processor. This mechanism specifies the phonetic form of the utterance, represented in Sound Level representations, which are then sent as instructions to the articulators.

Because more specific information about how the processes in the boxes in Figure 1 operate is needed in order to make specific predictions about actual speech production facts, it became necessary in my previous work to elaborate on the internal workings of the boxes. In particular, I focused my attention on the operations of the Syntactic Processor and sought answers to two types of questions: (1) What information is manipulated by the Syntactic Processor? and (2) What operations are carried out in that component? Let us consider each of these questions in turn.

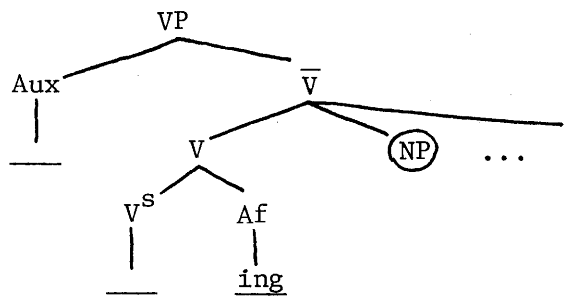
The types of information computed by the Syntactic Processor fall into three categories -- the Functional Level (FL) representations that are input to the Syntactic Processor, the positional frames themselves, and the structure of the positional frame store. Considering first the information contained in FL representations, Garrett is not entirely clear about what he means in using the phrase "grammatical relations" to indicate the kinds of notions included in those representations. Given the role that the Semantic Processor and these FL representations appear to be playing in Garrett's system, I have simply assumed that these representations include all of the linguistically relevant semantic relations that are to hold among the constituents of the ultimate utterance. Thus, to take some examples involving the meanings associated with English verb forms, we may assume that the relevant semantic notions are represented in FL representations as sequences of basic V notions, as in (1).

(1) V form	sequences of semantic notions
a. is V+ing	(indicative,active,durative,present,sing-3)
b. V+ed	(indicative,active,nonspecific,past)
c. was V+ing	(indicative,passive,nonspecific,past,sing)

Two comments about such sequences of semantic notions are in order. First, the use of these sequences here is intended solely as a convenient notational device. In the absence of a more fully articulated theory about the representations required in semantic processing, some sort of representation is needed to fill in for the presently unknown formulas. The sequences of notional categories in (1) are intended to serve that purpose in the case of grammatical marker meanings. Second, in my earlier work, properties of grammatical marker production were illustrated solely using V forms, VPs, and V semantic notions. There is, however, nothing special about focusing on these types of elements. Other items, for example, N forms, APs, etc., could just as easily be employed in illustrating how grammatical markers are produced in the model.

Turning next to properties of the positional frames, I have assumed that these are structural fragments of syntactic surface structures of the sort given in (2). Such fragments are assumed to consist of maximal phrases of major lexical categories expanded down to the minimal stem level and containing (at least) inflectional affixes already in place. These fragments are also assumed to contain indications of where other structural fragments are to be attached (in (2), the circled NP node), as well as slots where function words and the head lexical stem are to be inserted.

(2)



Finally, concerning the storage of information in the Syntactic Processor, the Lapointe (1984) paper assumes that there are three kinds of stores: one containing notion sequences, one containing fragments, and the third containing function words. Notion stores are assumed to be two-dimensional arrays organized according to the semantic markedness of the basic notions contained in the sequences. The fragment and function word stores are assumed to be unordered sets, each member of which is connected to the appropriate cell or cells of the notion stores. Thus, in the case of (1a), the Aux is and the VP fragment in (2), for example, would both be connected to the cell in the V notion store containing the sequence (indicative,active, durative,present,sing-3), or more properly, whatever that sequence corresponds to in the actual semantic representation. The purpose of the notion stores and connections is to serve as mediators between the incoming semantic notions in FL representations and the accessing of fragments and function words, a process that we will return to presently.

Turning now to the operations carried out by the Syntactic processor, that mechanism is assumed to include two major subcomponents besides the three types of stores just described -- a Control mechanism and a Stem Inserter, as shown in Figure 2 (next page). After accepting FL representations from the Semantic Processor, the Control then divides this information into parts which will be used for accessing fragments and function words and for keeping track of which lexical stem is to be inserted in which slot. Assuming that the information needed for fragment and function word accessing is contained in sequences of notions of

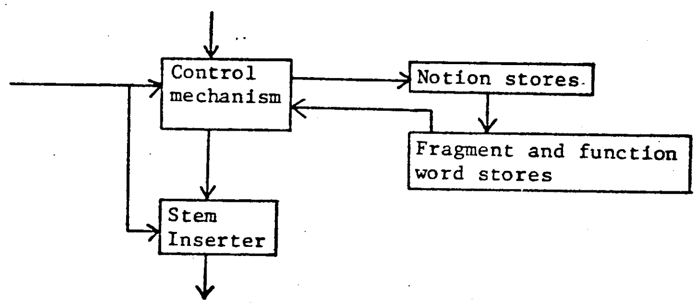


Figure 2. Organization of Syntactic Processor, after Lapointe (1984).

the sort in (1), the Control mechanism then takes these sequences and searches through the notion stores until it finds a match with the sequence from the input FL representation. [1] The cell with the matched sequence is then activated, which causes the fragment and function words connected to that cell to be automatically returned to the Control mechanism. That subcomponent then combines the fragments and function words and gives the combined structures to the Stem Inserter which, under the direction of the Control mechanism, inserts the previously selected lexical stems into the appropriate fragment slots.

That, in a nutshell, is the model of grammatical marker production proposed in the papers cited above. Much of the rest of those articles was devoted to exploring the specifics of the accessing operations in the domain of V forms, and to showing how such a system can account for a wide range of basic facts about V form production in the case of agrammatic aphasics. Since the issues of how the Control mechanism actually combines fragments and function words and directs the operations of the Stem Inserter were not directly related to the immediate concerns of that earlier work, nothing specific was said about those processes there. I would now like to take up these issues in the remainder of this paper.

2. The Combination System

There are three questions that we must answer about the fragment combination process: (1) What contribution do the FL representations make to the process? (2) How does the Control mechanism actually combine the fragments and function words? and (3) How does the Stem Inserter operate? Taking up each of these questions in turn, let us consider first the simple English sentence in (3).

(3) Mary kissed the boy.

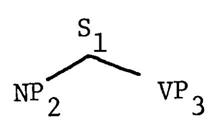
What sorts of information can we expect a FL representation for such a sentence to contain? Minimally, we need to be able to specify the propostional content of the statement to the effect that an individual named Mary and some contextually definite individual in the class of boys stand in the kissing relation, with Mary as the kisser and the boy as the kissee. In addition, the FL representation must specify that this event took place in the past. For the moment, we can represent this information along the lines in (4); again, I am not attempting to make any special claims about the actual representations used in FL structures by adopting this notation.

(4) [indicative,active,nonspecific,past] -- KISS([Ø] -- MARY, [def] -- BOY)

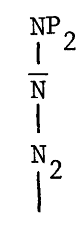
In this quasi-logical structure, the notional categories indicating that the event took place in the past and that the boy is contextually definite are appended to the left of the predicate and class symbols KISS and BOY, respectively. For reasons of convenience, I will assume that elements without notional categories are specified as having null ones, as is the case with the name MARY in (4). As I have already observed, the notional categories appearing in the brackets are used in accessing fragments and function words. The propositional content of a FL representation and the relation between that and the appended notional categories will be used in guiding fragment combination and stem insertion.

Continuing with this example, let us assume that at the point after fragments and function words have been accessed, but before combination proper has begun, the Control mechanism is looking at the information in (5).

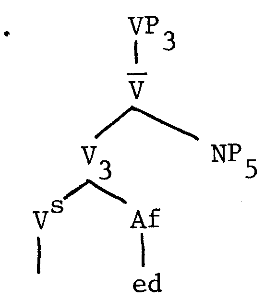
(5) a.



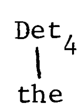
b.



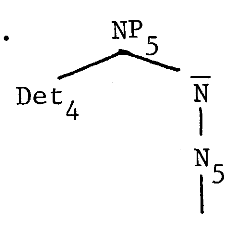
c.



d.



e.



f. $H(NP_2) = Lex(MARY)$

g. $H(VP_3) = Lex(KISS)$

h. $H(NP_5) = Lex(BOY)$

The root node of each fragment here bears an index which in effect will determine how the fragment attachments are to be processed, and there is a series of statements in (5f-h) relating the <u>heads</u> of the various phrases (represented by the notation '$H(XP_i)$') to the lexical realization of one of the constituents in the FL representation (represented by 'Lex(Q)').

In order to produce this set of structures and statements, I will assume that there is a system of "correspondence rules", a reflection of the grammar of the particular language involved, which the Control mechanism employs. These rules are assumed to contain two parts; one, the <u>search command</u>, directs the Control mechanism to access fragments and function words by searching through particular notion stores, while the other, the <u>indexing rule</u>, indexes fragments and determines head equations. Every correspondence rule must at least have an indexing rule, but it need not have a search command. For English then, some of the correspondence rules might look like those in (6).

(6) <u>Some English correspondence rules</u>

a. $[s_i]$ -- R($[s_j]$ -- X,...) ⇒ (i) search S/VP store for $[s_i]$
search NP store for $[s_j]$

(ii) $[s_i]$- S_q , $[s_j]$-NP_{q+1} , $[s_i]$-VP_{q+2}

NP_{q+1} (Aux) $[s_i]$-VP_{q+2}

$H(NP_{q+1})$ = Lex(X); $H(VP_{q+2})$ = Lex(R)

b. ...R($[s_j]$ -- X, $[s_k]$ -- Y,...) ⇒ (i) search NP store for $[s_k]$

(ii) $[s_i]$-VP_q , $[s_k]$-NP_{q+1} , $H(NP_{q+1})$ = Lex(Y)

$\bar{V}$

V_q NP_{q+1}

c. ...(..., [def] -- X,...) ⇒ [def]-NP_{q+1} , [def]-Det_q

Det_q $\bar{N}$

N_{q+1}

___

d. ...(..., [Ø] -- X,...) == [Ø]-NP_q , $H(NP_q)$ = Lex(X)

$\bar{N}$

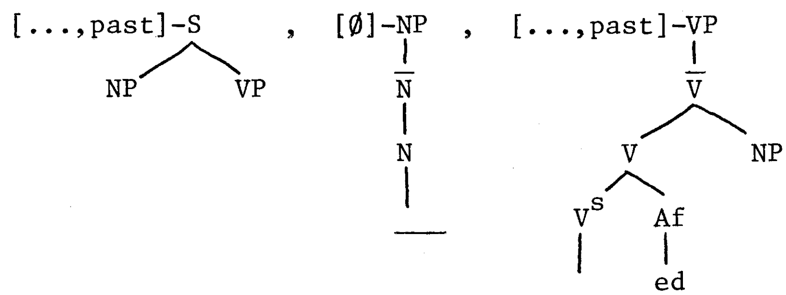
N_q

___

In these rules, the expression before the arrow contains a portion of the FL representation; R stands for a predicate, X, Y, etc. stand for class symbols and names, and s_i in brackets stands for notion sequences appended to the other elements. I am assuming that when a fragment or function word is activated during access, it is returned to the Control mechanism with a "flag" consisting of the notions that were used to activate that fragment or function word's cell. Some such device as these flags is needed so that the Control mechanism can keep track of which accessed fragment or function word corresponds to which piece of the FL representation. Also, in a slight departure from my earlier work, I am assuming here that function words are not stored as simple lexical or morphophonological items but rather are stored as very simple fragments, as in the case of (5d).

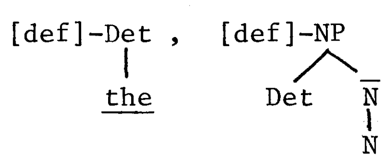
Let us now walk through the correspondence process for the FL representation in (4). First we have the rule in (6a) applying to the portion of the FL represention given in (7). (6a) directs the Control mechanism to search the S/VP store for the notion sequence [indicative,active,nonspecific,past], and to search the NP store for the sequence [∅]. The fragments in (8) are returned by this process.

(7) [...,past] -- KISS([∅] -- MARY,...)

(8)


The indexing rules of (6a,d) then apply to insure that the S-node is indexed; the lexical head and the root node of the NP fragment both have the same index, and that index is the same as the one assigned to the NP in the S fragment and is one more than the index for the S-node itself; the indices for the two VP nodes are the same, and are two more than that for the S-node; and the head statements for the NP and the VP are added. This results in (5a,b,f,g) and part of (5c). Next, (6b,c) apply to the portion of the FL representation given in (9), and the search command of (6b) leads to the accessing of the fragments in (10).

(9) ...KISS([∅] -- MARY, [def] -- BOY)

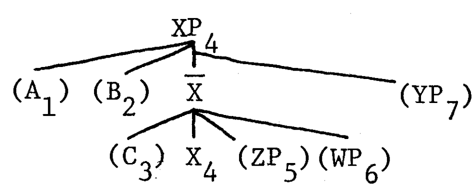
(10)


The indexing rule of (6c) insures that the lexical head and root nodes of the NP fragment will have the same index, and that that index will be one greater than that for the Det in the NP fragment and in the Det fragment. The indexing rule in (6b) will insure that the index on the root node of the NP fragment will be the same as that on the NP node in the VP fragment in (8), and the head statement for the NP is added. This results in the rest of (5c), as well as (5d,e,h). Finally, once all the necessary nodes have been indexed, the notion sequence "flags" can be eliminated by the general convention in (11). After this convention applies, we will be left exactly with the fragments and head statements in (5).

(11) Convention. After fragments have been indexed, delete all notion sequence flags attached to fragment root nodes.

What I would like to propose now concerning the actual combination of fragments is that there isn't any; that is, there is no actual attachment *per se*. Instead, what happens is that the fragments are simply arranged in a linear order determined by the indexing on the root nodes of the fragments. In fact, if the indexing is set up correctly, the ordering principle required can be stated in an almost perversely simple fashion. By "correctly" here, I mean that the correspondence rules can be set up in such a way that if the particular grammar in question has a PS tree like (12), then the indexing will be as indicated there.

(12)



Thus, the correspondence rules can be made to reflect an indexing algorithm which starts to the left of the $\bar{X}$ skeleton of a phrase and indexes each constituent except the $\bar{X}$ phrase nodes themselves from left-to-right at each $\bar{X}$ level from the top level down; when the left side is completed, it then indexes the lexical head of the phrase and then moves to the right side of the $\bar{X}$ skeleton, where it proceeds again from left-to-right but now moves from the lowest $\bar{X}$ level to the top; and finally, it assigns the root node the same index as that of the lexical head. [2] The indexing rules of (6) conform to such an algorithm. Assuming that this is always the case, the ordering principle can be given as in (13).

(13) Fragment Ordering Principle

Order fragments from left to right in ascending order of indices on the fragments' root nodes.

The reason why we can get away with a move like this is that there is an algorithm which will uniquely derive these sequences of fragments from arbitrary PS trees, and vice versa. We can visualize actual attachment lines linking coindexed nodes, if we choose, and there is a way to do that such that the lines will not cross. This representation of fragment sequences is therefore equivalent to representations in terms of the more familiar sorts of PS trees. All that we have done is to take hierarchical depth, below the level of immediate dominance between mother and daughter nodes, and translated it into the linear dimension in such a way that the order of terminal elements is preserved. With a certain amount of foresight, I have arranged the fragments in (5a-e) so that they are already in the order stipulated by the Ordering Principle in (13), and hence in this example at least, nothing more needs to be said.

Notice though that this will not always be the case. Even if we were to <u>require</u> that the correspondence rules apply in a fixed order, so that the Control mechanism would first have to try to access fragments that generally occur before other fragments, such a requirement would not necessarily guarantee that the earlier fragments would be <u>returned</u> first. According to the access system developed in the papers cited above, the more semantically complex the notion sequences associated with fragments are, along dimensions made explicit in those papers, the longer it will take for the corresponding fragments to be retrieved. Thus, it is quite possible that the Control mechanism would start searching for a fragment F_1 some time before it begins to look for a fragment F_2, but F_2 is retrieved more quickly than F_1, assuming that F_2 is sufficiently simple compared to F_1 in terms of notions sequences. Therefore, while there may be reasons for wanting to have the correspondence rules ordered with respect to one another, at least this idea of order of fragment access paralleling order of fragment combination as determined by (13) will not count as such a reason.

We can also consider various ways of making the system somewhat neater. For example, we might entertain the possibility of having fragments stored with the full range of optional constituents for phrases of the types in question, and with the indexing relations already specified in the fragments. When the correspondence rules applied, whatever optional constituents inside a fragment were not coindexed with any nodes outside the fragment would simply not be realized. Such a modification would permit a simplification in the statement of the correspondence rules and would basically eliminate the need for a mechanism that would otherwise be required to determine when optional constituents of fragments must or must not be present.

Finally, I will have a good deal less to say about the way in which lexical stems are inserted into the appropriate slots in the ordered fragments, largely because the insertions interact heavily with the syntax/phonology interface which I am trying not to discuss here. In rough outline, however, the process works like this. First, items are output from the lexicon and input to the

Control mechanism in the form given in (14), where Q is, as before, a constituent of FL representations, and P is a paired morphological/phonological structure for the stem.

(14) Lex(Q) = P

These inputs, together with the head statements produced by the action of the correspondence rules, will determine which stem is to be inserted into which slot. Thus, to finish off the example we have been working with, assuming all goes well, the lexicon will output the stems in (15), where standard orthographic representations are being used instead of the paired morphological/phonological structure.

(15) a. Lex(MARY) = Mary
b. Lex(KISS) = kiss
c. Lex(BOY) = boy

Given the head statements in (5f-h), Mary will be inserted in the slot in (5b), kiss in the slot in (5c), and boy in the slot in (5d), as desired. As the stems are inserted, the phonological structures of the stems are read off in the order determined by the sequence of fragments and sent as input to the Phonetic Processor.

3. Speech Error Data

Let us now turn to some speech error data to see how this model fares against some facts. Potentially, there are four places in the Syntactic Processor where errors could occur. These are listed in (16).

(16) a. initial analysis of FL representations by correspondence rules
b. assignment of indices and head statements by correspondence rules
c. ordering of fragments
d. stem insertion

It is unlikely that errors would arise in the case of either (16a) or (16c). The only potential problem in the analysis of the FL representation is for the Control mechanism to try to search for a notion sequence in the wrong notion store; for example, it might try to look for the [indic.,act.,nonspec.,past] sequence of (4) in the NP store. If this happens, one of two events will occur: either the search will proceed until the Control mechanism reaches the end of the store without locating the required notion, because that notion sequence is not in the store being searched, or the Control mechanism catches the error before the search begins. In either case, the whole process should stop until the Control mechanism can start over and get the analysis of the FL representation right. This should result in an interruption of speech, but not an error *per se*. In the case of ordering (16c), given the simplic-

North East Linguistics Society, Vol. 15 [1985], Iss. 1, Art. 18

the N head was accessed before the N side and hence was inserted into the only open N slot. The sorts of errors in (19) - (24) can apparently be explained then in terms of processing breakdowns which are separate from the ones of immediate concern in the present discussion.

Another type of error belonging in this first class of syntactic errors, syntactic blends, is relevant, however. Some examples are given in (25).

(25) Syntactic blends
a. T1: OK, turn on the light
   T2: OK, turn the light on
   E: OK, turn on the light on
b. T1: No, I think you're right
   T2: No, I think that's true
   E: No, I think you're true

Here we have cases where fragments that are very close structurally and are both semantically identical to, or very close to, the specification given by the FL representation have been accessed for the same utterance. If each is assigned the same index, and both fragments remain through the ordering stage, pieces of each of the fragments will be read out into the Phonetic Processor. We thus have some evidence for a breakdown in the operation of the indexing rules, albeit one that results from a prior error involving the accessing of two fragments for a single FL representation.

Turning now to the second types of syntactic speech error data, we find that by and large this type seems to involve substitutions of words or word stems. Errors in this class have been studied extensively by Garrett (1975, 1980), from whom we have the examples in (17) and (26).

(26) Full word exchanges
a. T: ...and give my back a hot bath
   E: ...and give my bath a hot back
b. T: I have to fill up my car with gas
   E: I have to fill up my gas with car

Garrett's account of these errors runs along the following lines. Full word exchanges behave differently from stranded morpheme errors. The first type nearly always involves items of the same lexical category, the words head phrases that typically play the same or very similar semantic roles, and the exchanging items need not be in the same clause. On the other hand, stranded morpheme errors often involve items from different lexical categories, and hence involve items playing different semantic roles, the items nearly always appear in the same clause, and the items are often phonologically similar and appear in phonologically similar contexts. In addition, the same sorts of constraints as are found in the case

Let us now turn to further syntactic speech error data to see whether these consequences of the model are in fact borne out. It is reasonable to divide the data that is usually discussed in this area into two parts. The first set has been studied in some detail by Fay (1980) and more recently by Stemberger (1982). The errors in this first set consist mainly of syntactic misproductions that are more complex than simple word substitutions. In this class we find errors like those listed in (19) - (24), from the works just cited.

(19) Subject/Aux Inversion
a. T: Uh-oh, where is it?
E: Uh-oh, where it is?
b. T: We'll see then how things are going
E: We'll see then how are things going

(20) Do-Support
T: It just does not grab you
E: It just not grabs you

(21) Complement restrictions
T: We can hardly avoid visiting my grandfather
E: We can hardly avoid to visit my grandfather

(22) Passive
T: You can't do that to me
E: I can't be done that to

(23) WH-Movement
T: Linda, which ear do you talk on the phone with?
E: Linda, do you talk on the phone with which ear?

(24) "Radical" extensions
T: Did you hit the right side of your head against the wall?
E: Did you hit your right head against the wall?

Stemberger argues, I think reasonably persuasively, that all of these cases can be explained either in terms of a mistake in accessing the fragments involved or as a combination of accessing and insertion errors. Thus, in terms of the present system, the error in (19b) arises from the selection of the S fragment that is appropriate for matrix rather than embedded questions. The other errors in (19) - (21) can be explained as variations on the same theme. In the case of (22), the error was reported to be uttered in a context in which the speaker was the topic of the conversation. As a result, in the construction of the FL representation, the 1st person pronoun was placed in first argument position, and subsequently the Syntactic Processor accessed a passive VP fragment containing the extra Aux node for be appropriate for this FL representation. In the WH-Movement case in (23), either of two things could have happened. The fragment appropriate for simple yes/no questions could have been accessed, forcing the wh-phrase which ear into the only possible NP slot, the one at the end of the S, or an S fragment with a null COMP and an NP fragment with full NP structure were accessed. Last, in the case of "radical" extensions like (24), the fragments needed for of your head would simply not have been accessed, your would have been accessed in place of the Det the, and ity of the ordering principle in (13), we should expect the sequencing of fragments to run in a completely automatic way, much as the return of activated fragments is fully automatic during notion store search. As a result, we should expect no problems to arise here, at least in the case of normal speakers.

This leaves the mechanisms in (16b,d) as possible places where errors can be generated. Taking stem-insertion (16d) first, the most likely error that can occur here involves the misplacement of stems. This will in general lead to the kinds of errors that Garrett calls "stranded morpheme errors"; some illustrations are given in (17) where 'T' = target and 'E' = error.

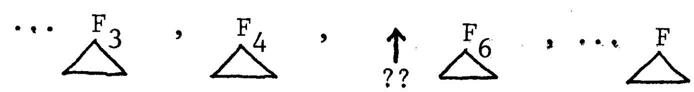
(17) Stranded morpheme errors (stem exchanges)

a. T: It just started to sound
   E: It just sounded to start

b. T: talking Turkish
   E: Turking talkish

Looking finally at (16b), we find that there are several sorts of potential errors that can arise in the operation of the indexing rules. First, it may be that no index is assigned to some fragment; this is the situation depicted in (18).

(18)



In such cases, when the fragments are ordered, there will be a gap between two of the fragments; in (18) the gap is between fragments indexed 4 and 6. Presumably in these cases the process stops until the Control mechanism can locate the unindexed fragment, index it, and make sure that it belongs in the sequence where the missing fragment belongs. This, then, will be another instance where no error actually occurs, but instead we should expect to find an interruption in the speech flow. Second, two fragments might be assigned the same index. This will occur, for example, when two fragments of the same root categories, with nearly identical structures, whose semantic properties correspond very closely to the FL representation, are accessed at the same time. If the Control mechanism does not catch the error, pieces of both fragments may be read out to the Phonetic Processor, leading to blend errors, which I will have more to say about shortly. Third, two fragments may be assigned the opposite indices from the ones they should receive. Again, this will be more likely to occur to the extent that the fragments involved are structurally alike. The result of such misassignment will be a kind of fragment exchange or anticipation, about which more later. Finally, the Control mechanism may assign the wrong head statement to a fragment. As was the case with stem insertion errors, this type of misassignment should also result in stranded morpheme errors of the sort in (17).

of stranded morpheme errors are also exhibited in the case of simple sound exchange errors, examples of which have not been given here. To account for these differences, Garrett assumes that the full word exchanges occur as a result of the assignment of incorrect semantic roles to items during the construction of FL representations by the Semantic Processor, while he assumes that both the stranded morpheme and sound exchange errors result from phonological material being incorrectly inserted during the construction of the Positional Level representation.

However, the present production system requires us to adopt a slightly different interpretation from the one Garrett gives for these errors. In addition to allowing stranded morpheme errors to arise from the manipulation of the phonological material of inserted stems, the present system also permits such errors to result from misassignment of head statements to fragments. Within the present model then there are two sources for these errors; we should therefore expect that there will be slight differences in the errors produced by the failure of those separate mechanisms. In fact, as the system is arranged, we should expect that stranded morpheme errors produced from the indexing rules should not involve items that are phonologically similar. Furthermore, we might expect that these same errors should involve items whose head statements were determined by the same correspondence rule. As far as the first prediction is concerned, there are numerous examples of stranded morpheme errors in which the two items are not especially phonologically similar or in similar environments; some instances from Garrett (1975) are given in (27).

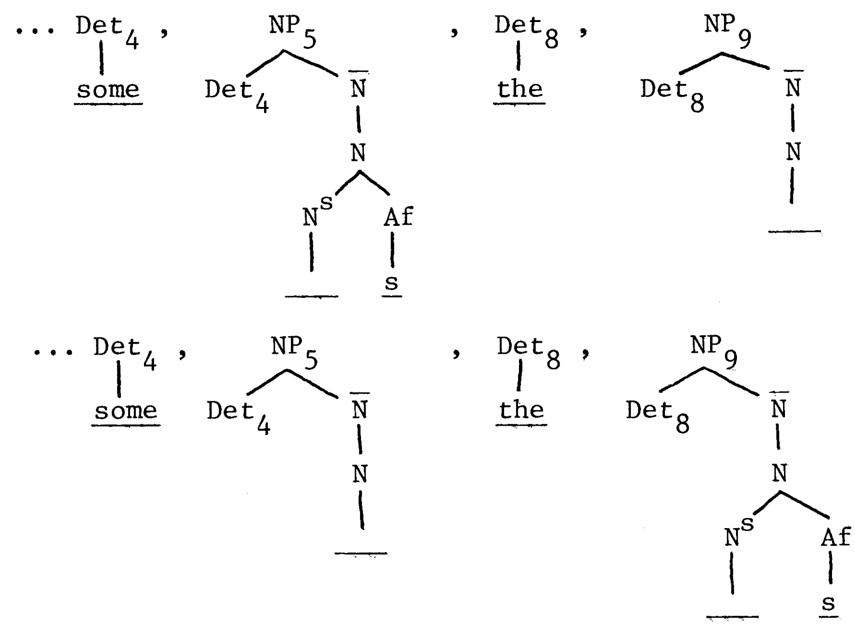
(27) Phonologically dissimilar stranded morpheme errors
a. T: Make it so that the tree has less apples
   E: Make it so that the apple has less trees
b. T: ... but the two's cleaner
   E: ... but the clean's twoer
c. T: OJ is just thirst-quenching
   E: OJ is just quench-thirsting

As for the second prediction, I have unfortunately not yet had a chance to work out all of the correspondence rules required in full enough detail to be able to report definitely on the issue here. However, to the extent that we can show that all of the phonologically dissimilar stranded morpheme errors also have head statements determined by the same correspondence rules, then the present system will gain further support.

Looking now at full word exchanges, within the present system such errors should arise when indices are assigned to the wrong fragments by the Control mechanism in the Syntactic Processor. Thus, to produce a hypothetical full word error like (28), the Control mechanism would have to break down in the way indicated in (29); that is, everything would be the same in the correct and errorful versions of (29) except that in the error case, the two NPs would have the opposite indices, and because this leads to the interchange of the two NP fragments when they are ordered, the head Ns of those fragments, including any affixes attached to them, will also be interchanged.

(28) T: Some boys saw the girl
E: Some girl saw the boys

(29) T: E:



Therefore, where Garrett assumes that these errors result from a malfunction in the construction of FL representations in the Semantic Processor, the present model assumes that they arise from malfunctions of the Control mechanism of the Syntactic Processor. What sorts of errors should we find then in the present model if FL representations are incorrectly interchanged? Consider such an exchange in (30).

(30) T: ...SEE([indef] -- BOY, [def] -- GIRL)
E: ...SEE([def] -- GIRL, [indef] -- BOY)

Here we see that the two arguments of the predicate SEE have been switched. This error should result in surface Ss in which full phrases have been interchanged. From the switch in (30) then, assuming that no other errors occur, we should obtain a surface error like (31).

(31) T: A boy saw the girl
E: The girl saw a boy

Such errors do in fact occur; some examples taken from Fromkin (1973) are given in (32).

(32) Full phrase exchanges

a. T: I have to smoke a cigarette with my coffee
   E: I have to smoke my coffee with a cigarette

b. T: a model of performance
   E: performance of a model

c. T: If you'll stick around you'll meet him
   E: If you'll meet him you'll stick around

Although Garrett has not focused on these errors as a type of particular importance to his research, he does mention in his 1980 paper that such errors exist and suggests that they also arise, like full word exchanges, as a result of a switch error in the construction of FL representations. However, since neither Garrett nor myself have specified what we mean by a FL representation in sufficient enough detail, determining which approach to full word and full phrase errors is correct is a matter that will have to be postponed until the internal structure of FL representations can be worked out more explicitly. As a summary of this admittedly inconclusive discussion of stem, word, and phrase exchanges, I offer the table in (33) which describes the differences between Garrett's proposals and the ones offered here.

(33)

Error type	Garrett's system	Proposed model
a. full phrase	FL construction	FL construction
b. full word	FL construction	fragment indexing
c. stranded morpheme (stem exchanges)	stem insertion	(i) head statement assignment (ii) stem insertion

4. Conclusion

To sum up then, we have found that in specifying the mechanisms responsible for combining structural fragments selected during syntactic processing, it is not necessary for the system actually to combine fragments. Instead, assuming that the fragment indexing mechanism is judiciously arranged, all we need to do is order the fragments according to their indices, and the combination problem is readily solved. In addition, we have seen that there is some initial support for the model in terms of syntactic speech errors, although further research needs to be undertaken to determine whether Garrett's system or the model proposed here offers the more appropriate explanation for some of the facts. Although it is interesting at this point to speculate about how such a system of fragment indexing and ordering might interact with other processing issues -- for example, how such ordered fragments would interact with the processing of phonological phrases, or to what extent the representations and operations of the present model can be carried over into the domain of syntactic comprehension -- I have not examined these areas sufficiently to offer anything more than a few quite rough and general comments. These sorts of questions must, alas, await further research.

FOOTNOTES

[1] The way in which the search is conducted is of considerable interest in its own right and a good deal of discussion was devoted to this issue in Lapointe (to appear a, 1984). I will not be concerned with this topic here, however.

[2] Lyn Frazier pointed out to me after the presentation that I had omitted a point which might confuse the reader, and which I wish to clarify here. I am assuming that S is an exocentric phrase with nonlexical head VP. If there are such exocentric phrases, then something will have to be added to the algorithm sketched in the text to accomodate indexing within such phrases. This can easily be accomplished, however, by simply treating the phrasal head of the exocentric phrase as if it defined an $\bar{X}$ skeleton, and then proceeding as in the text.

REFERENCES

Bock, K. (1982) "Toward a Cognitive Psychology of Syntax: Information Processing Contributions to Sentence Formulation," Psychological Review 89, 1-47.

Cooper, W. and J. Paccia-Cooper (1980) Syntax and Speech, Harvard University Press, Cambridge, MA.

Fay, D. (1980) "Transformational Errors," in V. Fromkin, ed., Errors in Linguistic Performance, Academic Press, New York.

Fromkin, V. (1973) Speech Errors as Linguistic Evidence, Mouton, The Hague.

Garrett, M. (1975) "The Analysis of Sentence Production," in G. Bower, ed., Psychology of Learning and Motivation, Vol. 9, Academic Press, New York.

Garrett, M. (1980) "Levels of Processing in Sentence Production," in B. Butterworth, ed., Language Production, Vol. 1: Speech and Talk, Academic Press, London.

Lapointe, S. (1983) "Some Issues in the Linguistic Description of Agrammatism," Cognition 14, 1-39.

Lapointe, S. (1984) "Simplifications to a Theory of Grammatical Marker Production in Aphasic Speech," unpublished paper, Wayne State University, Detroit, MI.

Lapointe, S. (to appear a) "A Theory of Verb Form Use in the Speech of Agrammatic Aphasics," Brain and Language.

Lapointe, S. (to appear b) "Markedness, the Organization of Linguistic Information in Speech Production, and Language Acquisition," in F. Eckman et al., eds., Markedness, Proceedings of the University of Wisconsin-Milwaukee Symposium, Plenum, New York.

Stemberger, J. (1982) "Syntactic Errors in Speech," Journal of Psycholinguistic Research 11, 313-345.