Lambek Calculus and Preposing of Embedded Subjects*
Richard T. Oehrle & Shi Zhang
University of Arizona
in memory of Sunseek Oh

0. The following syntactic paradigm in English is well-known:

1) Kim said that Sandy likes chimichangas.
2) Kim said Sandy likes chimichangas.
3) *I wonder who Kim said that _ likes chimichangas.
4) I wonder who Kim said __ likes chimichangas.
5) I wonder what Kim said that Sandy likes __.
6) I wonder what Kim said Sandy likes __.

A variety of accounts of this paradigm have been offered in the literature. We mention Bresnan's (1972) Fixed Subject Constraint and the *that-trace filter of Chomsky & Lasnik (1977) and subsequent work in the GB tradition. These two accounts share the view that the explanation of the oddity of (3) lies in prohibiting an element of a certain sort -- namely, the complementizer that -- from preceding a gap of a certain kind. In this paper, we investigate the problem from a somewhat different angle. We assume an analytic framework in which there are no gaps, and hence, no gaps of the sort required by the accounts of either Bresnan or Chomsky & Lasnik.

We first review the properties of two categorial systems, the Associative Syntactic Calculus AL (Lambek, 1958) and the NonAssociative Syntactic Calculus nAL (Lambek, 1961), and show how they are adapatable to the analysis of simple sentences. We then show, following Steedman (1985), how categorial systems of this kind can treat discontinuous dependencies of the sort found in English topicalization and wh-movement in simple clauses. It is not altogether obvious how to treat the paradigm above within this framework of assumptions. The remainder of the paper is devoted to showing how this can be done.

1.0 Associative and NonAssociative versions of the Lambek Calculus

In general, categorial grammar is an approach to grammatical analysis in which the linguistic properties of a complex expression are taken to be a function of the corresponding properties of its component parts.1 In their pure form, the Lambek Calculi exemplify a systematic approach to categorial analysis: that is, allowable principles of type-assignment are assumed to apply to expressions of every type.

Let a countable set P of primitive types be given. The full set C of categories is the least set containing P and, when it contains x and y, contains (x/y), (y\x), and (x.y). (In what follows, we exhibit types in a way that ignores unnecessary parentheses.) The arrow x -> y means that any expression of type x is also assigned the type y. The following axioms and inference rules characterize the Associative Calculus AL. The nonAssociative Calculus nAL lacks the associative rules A2 and A2'.

Among the theorems common to AL and nAL are the following:

apply: \( x/y.y \rightarrow x \) ; \( y.y/x \rightarrow x \)
lift: \( x \rightarrow y/(x\backslash y) \) ; \( x \rightarrow (y/x)\backslash y \)

There are also derived rules of inference valid in both systems:

R4. \( x \rightarrow x' \) \( y \rightarrow y' \)
\[ xy \rightarrow x'y' \]
R5. \( x \rightarrow x' \) \( y \rightarrow y' \)
\[ x/y' \rightarrow x'/y \]

If we regard elements of type \( x/z \) as functors with domain \( z \) and co-domain \( x \), R5 reflects the fact that for any function \( f: D \rightarrow C \), if \( D' \) is a subset of \( D \) and \( C' \) is a superset of \( C \), there is a unique function \( f': D' \rightarrow C' \) such that \( f'(d) = f(d) \) for all \( d \) in \( D' \).

In addition, a number of other interesting theorems which are not derivable in nAL hold in AL:

compose: \( (x/y).(y/z) \rightarrow x/z \) ; \( (z\backslash y).(y\backslash x) \rightarrow z\backslash x \)
divide: \( x/y \rightarrow (x/z).(y/z) \) ; \( y\backslash x \rightarrow (z\backslash y)\backslash(z\backslash x) \)
swap: \( (x\backslash y)/z \leftrightarrow x(y/z) \)
Curry: \( (x/y)/z \leftrightarrow x/(z,y) \)

Both AL and nAL are decidable. In addition, AL has the property of structural completeness (Buszkowski, 1988), which means that if there is a proof of the validity of an arrow relative to one bracketing, there are proofs of the arrow for every well-formed bracketing.

2. Linguistic applications.
If we are given a vocabulary \( V \), assign each element of \( V \) to a finite, non-empty set of types, and assume that a sequence \( v_1,...,v_k \) of words (on a fixed bracketing if we are working in nAL) is assigned to the type \( z \) (a fact we
represent by the arrow \( v_1, \ldots, v_k \rightarrow z \) just in case there are types \( x_i \) assigned to each \( v_i \) \((1 < i < k)\) such that \( x_1 \ldots x_k \rightarrow z \) (on the same bracketing, if we are working in nAL) is a theorem. For example, relative to following assignments,

\[
\begin{align*}
\text{Kim} & \rightarrow \text{np} \\
\text{put} & \rightarrow (\text{np}\text{s}/\text{pp})/\text{np} \\
\text{Nim} & \rightarrow \text{np} \\
\text{near} & \rightarrow \text{pp}/\text{np} \\
\text{Zim} & \rightarrow \text{np}
\end{align*}
\]

the sequence of words \textbf{Kim put Nim near Zim} is assigned to the type \( s \), since the sequence of types \( \text{np}.((\text{np}\text{s}/\text{pp})/\text{np}).\text{pp}/\text{np}.\text{np} \rightarrow s \) is a theorem of AL. Here is a proof, which uses a rule of inference (easily derivable from R4) that allows substitution of a type \( z \) for a sequence of types \( y_1 \ldots y_k \) just in case \( y_1 \ldots y_k \rightarrow z \) is a theorem. Each step is annotated with the name of the theorem involved.

\[
\begin{array}{cccc}
\text{Kim} & \text{put} & \text{Nim} & \text{near} & \text{Zim} \\
\text{np} & (\text{np}\text{s}/\text{pp})/\text{np} & \text{np} & \text{pp}/\text{np} & \text{np} \\
& & & (\text{apply}) & \\
& & & \text{pp} & \\
& & & (\text{apply}) & \\
& & & (\text{np}\text{s}/\text{pp}) & \\
& & & (\text{apply}) & \\
& & & \text{np}\text{s} & \\
& & & (\text{apply}) & \\
& & & s & \\
\end{array}
\]

In nAL, of course, this proof requires the bracketing \((\text{Kim}.(\text{put.Nim})\cdot(\text{near.Kim}))\). But in AL, proofs exist relative to other bracketings:

\[
\begin{array}{cccc}
\text{Kim} & \text{put} & \text{Nim} & \text{near} & \text{Zim} \\
\text{np} & (\text{np}\text{s}/\text{pp})/\text{np} & \text{np} & \text{pp}/\text{np} & \text{np} \\
& & & (\text{lift}) & \\
& & & \text{s}/(\text{np}\text{s}) & \\
& & & (\text{div}) & \\
& & & (\text{s}/\text{pp})/((\text{np}\text{s})/\text{pp}) & \\
& & & (\text{div}) & \\
& & & ((\text{s}/\text{pp})/\text{np})/((\text{np}\text{s})/\text{pp})/\text{np} & \\
& & & (\text{apply}) & \\
& & & (\text{s}/\text{pp})/\text{np} & \\
& & & (\text{apply}) & \\
& & & \text{s}/\text{pp} & \\
& & & (\text{apply}) & \\
& & & \text{s}/\text{np} & \\
& & & (\text{apply}) & \\
& & & s & \\
\end{array}
\]

This proof corresponds to one version of a left-to-right incremental parse. In AL, there are a variety of other proofs as well. The fact that categorial systems that respect the axioms of AL are structurally complete suggests that they offer an interesting framework in which to investigate such problems as
"non-constituent" conjunction (Steedman, 1985; Dowty, 1988; Oehrle, 1987), the bracketing paradoxes in morphology (Moortgat, 1988; Hoeksema, 1985), and the relation of syntactic structure and prosodic phrasing (Moortgat, 1989; Oehrle, 1988).

2.1 Wh-movement in AL.
The symmetry of the Lambek operators / and \ suggests a rule which replaces one by the other, a rule we call permutation:

\[
x/y \leftrightarrow y/x
\]

The Lambek Calculi as formulated above are permutation-free. That is, none of the axioms or rules of inferences permutes expressions. To account for alternations such as wh-movement or topicalization, an additional rule must be added. Consider first two possible proofs of the sentence Kim greeted Zim, assuming that greeted \(\rightarrow (np/s)/np\) and, as above, Kim \(\rightarrow np\) and Zim \(\rightarrow np\):

\[
\begin{align*}
\text{Kim} &\quad \text{greeted} &\quad \text{Zim} \\
np &\quad (np/s)/np &\quad np \\
\text{apply} &\quad np/s &\quad np/np \\
\text{apply} &\quad s &\quad s/np \\
\text{apply} &\quad s &\quad s/(s/np)
\end{align*}
\]

In the proof on the right, Zim is lifted to the category \((s/np)/s\). Now, to generate structures involving topicalization, we need only add a rule which permutes lifted types whose "co-domain category" is S. (This idea is based on Ades & Steedman (1982) and Steedman (1985).) In the simplest cases, we may formulate the rule as follows:

\[
\text{permutation-lifting into } s: np \rightarrow s/(s/np)
\]

To see how this works, consider:

\[
\begin{align*}
\text{Zim}, &\quad \text{Kim greeted} \\
np &\quad np \quad (np/s)/np \\
\text{as above} &\quad s/np \\
\text{permutation-lifting} &\quad s/(s/np) \\
\text{apply} &\quad s
\end{align*}
\]

Remarks. First, it is easy to arrange matters so that if we wish to provide an interpretation of such a language (as we should wish to do), the interpretation of simple cases of the kind illustrated here will be equivalent to the interpretations of the two derivations of Kim greeted Zim given earlier. Second, there are some wrinkles that have to be attended to. As stated, permutation-lifting allows multiple instances of (nested) topicalization, which seems forced in English. In addition, this formulation only works for right-
peripheral "gaps". These points, which are orthogonal to the issues we wish to address here, can be resolved by restating permutation-lifting as PL below (where X is a variable ranging over the union of the set of types and 0: if X matches 0, then y/X = X\y = y).

8) PL: np → (s[F]/X)/((s/X)/np)

For example, suppose that embedded questions are of the category s[q], which we shall abbreviate as q. Let who be assigned to the type (q/X)/((s/X)/np). To see the consequences of this type-assignment, we shall examine its application to some of the cases we began with. (We write "inst X" to mark the step which instantiates the variable X in a particular way.)

<table>
<thead>
<tr>
<th>I wonder</th>
<th>who</th>
<th>Kim</th>
<th>saw</th>
</tr>
</thead>
<tbody>
<tr>
<td>s/q</td>
<td>(q/X)/((s/X)/np)</td>
<td>s/np</td>
<td></td>
</tr>
<tr>
<td>q/(s/np)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

I wonder what (q/x)/((s/x)/np) Kim put (np\s)/pp/np there pp

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>(s/np)/np</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(inst X)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(q/np))/((s/np)/np</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>q/np</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>q</td>
<td></td>
</tr>
</tbody>
</table>

Having shown the extension of AL to AL+PL can account for simple cases of peripheral and non-peripheral "extraction" (without movement rules), we turn to the analysis of the paradigm mentioned at the outset.

3. Assumptions.
We shall suppose that the complementizer that is assigned the type c/s and that (1) and (2) are distinguished by lexically assigning said two different types: (np\s)/c and (np\s)/s. One proof of Kim said that Sandy likes chimichangas is given in (9) below. And a proof of Kim said Sandy likes chimichangas which simply removes that and replaces the type (np\s)/c with the type (np\s)/s is given in (10).
The permutation-lifting analysis of "wh-movement" discussed above can be straightforwardly applied to cases of non-subject extraction. Here are proofs of (5) and (6) above, assuming that what is assigned the type \((q/X)/(s/X)/np\) and wonder is assigned the type \((np\backslash s)/q\):

\[
\begin{align*}
\text{I} & \quad \text{what} \quad \text{Kim} \quad \text{said} \quad \text{that} \quad \text{Sandy} \quad \text{likes} \quad \text{chimichangas.} \\
\text{np} & \quad (np\backslash s)/q \quad (q/X)/((s/X)/np) \quad \text{s/q} \quad \text{-- (lift)} \quad \text{-- (apply)} \\
\text{np} & \quad (np\backslash s)/c \quad c/s \quad np \quad (np\backslash s)/np \quad np \\
\text{--- (swap)} \\
\text{np} & \quad (s/c) \quad \text{--- (apply)} \\
\text{s/c} & \quad \text{--- (apply)} \\
\text{s/s} & \quad \text{--- (lift)} \\
\text{s/(np\backslash s)} & \quad \text{--- (apply)} \\
\text{s/(np\backslash s)} & \quad \text{--- (apply)} \\
\text{s/(np\backslash s)} & \quad \text{--- (apply)} \\
\text{s} & \quad \text{--- (apply)} \\
\end{align*}
\]

\[
\begin{align*}
\text{72}
\end{align*}
\]
I wonder what Kim said Sandy likes chimichangas

--- (lift)

s/(np\s) s/(np\s)

------------------------- (compose)

s/q

------------------------- (compose)

s/s

--- (lift)

s/(np\s)

------------------------- (compose)

s/(np\s)

------------------------- (inst X) (comp)

q/(s/np) s/np

------------------------- (apply)

q

------------------------- (apply)

s

4. Subject "extraction" in the complementizerless case.
Consider now the extension of the account developed above to cases involving a "gap" in subject-position. Suppose we apply our strategy of assigning the permutation of a lifted type to the wh-phrase and compiling the material between the wh-phrase and the gap into a single constituent assigned to a type of the form (s/X)/np. This yields:

... who Kim said likes chimichangas

(q/X)/((s/X)/np) np (np\s)/s (np\s)/np

------------------------- (as above) ------------------------- (apply)

s/s np\s

At this point, no further reduction is possible. To resolve this problem, various nostrums have been suggested. (Our discussion here is based on remarks by Steedman (1987, section 3.2.2, pp. 421ff.). One possibility, following a suggestion in GKPS (1985), is to assign to said and similar verbs the category (np\s)/(np\s). This still requires that new forms of reduction be countenanced. Steedman (1987) suggests a different approach: assign said and similar verbs to the category ((np\s)/np)/(np\s). On this approach, extraction in such cases is assimilated to right-peripheral extraction, which is in itself unproblematic. A drawback here, however, is the prediction that we should allow structures in which the embedded subject is shifted to the right, as in *Kim said likes chimichangas Sandy. We prefer a theory which excludes such cases.

Within the general framework of AL+PL, in fact, extraction of the type that we have been discussing is completely unproblematic. We first prove the following lemma, which is simply a special case of the derived rule of inference R5, mentioned earlier in section 1:

Lemma 1. \( X/s \rightarrow X/(np.(np\s)) \)

proof: \( X \rightarrow X \)

\( np.(np\s) \rightarrow s \)

------------------------- R5

\( X/s \rightarrow X/(np.(np\s)) \)

Now note that the arrow \( X/(np.(np\s)) \rightarrow (X/(np\s))/np \) is an instance of Currying (theorem (6)), so that by the transitivity of the arrow, we have the
theorem $X/s \rightarrow (X/(np\backslash s))/np$. This proves Lemma 2:

**Lemma 2.** $X/s \rightarrow (X/(np\backslash s))/np$

The proof that who Kim said likes chimichangas $\rightarrow q$ is now completely straightforward:

... who
(q/X)/((s/X)/np)  Kim said likes chimichangas
(np/\backslash s)/s (np/\backslash s)/np
np

\[\text{(swap)}\]

\[\text{(app)}\]

\[s/s\]

\[\text{(inst \text{Lemma})}\]

\[(s/(np\backslash s))/np\]

\[\text{(apply)}\]

\[q/(np\backslash s)\]

\[\text{(apply)}\]

\[q\]

5. Subject "extraction" and complementizers.
In the system AL+PL, given the syntactic categories thus far assumed, there is a proof of *I wonder who Kim said that ___ likes chimichangas. Here is the crucial sub-proof that who Kim said that likes chimichangas is assigned type q. As in the case just discussed, the proof makes use of Lemma 2, which shows that the arrow c/s $\rightarrow (c/(np\backslash s))/np$ is valid.

who
(q/X)/((s/X)/np)  Kim said that likes chimichangas
(np/\backslash s)/c (np/\backslash s)/np

\[\text{(inst X)}\]

\[\text{(Lemma 2)}\]

\[(q/(np\backslash s))/((s/(np\backslash s))/np)\]

\[\text{(apply)}\]

\[q/(np\backslash s)\]

\[\text{(apply)}\]

q

It is just as clear that if it is impossible to assign the phrase Kim said that a non-product type of the form (s/X)/np, no proof will be available which detects a "gap" after that. Since it is necessary on independent grounds to allow the combination of Kim said into a single non-product type, our problem can be solved if we can find a way to solve a related problem: blocking the composition of said and that. We can achieve this goal by adopting as a
framework a Lambek system which is partially-associative.

From the present perspective, there are clear empirical constraints on the properties that a partially-associative syntactic calculus of the right kind may have. First, as just noted, we must require that the following inference step never be allowed:

\[
x \cdot (\text{that}.y) \\
\hline
(x \cdot \text{that}).y
\]

The strategy we will implement below is to ensure that certain elements, including the complementizer that, introduce brackets grouping them with their arguments.

A second point in the case at hand is the nature of the argument type in the lexical type-assignment of that. The type assumed above, namely c/s, is not the only possible type that could be lexically assigned to that: another possibility is \((c/(np\backslash s))/np\). Since c/s readily accommodates conjoined complements, however, and \((c/(np\backslash s))/np\) does not, we shall assume here that c/s is at least one type available to that. (In AL, note that c/s \(\rightarrow\)
\((c/(np\backslash s))/np\) is a theorem, derived by means of R5 and Currying.) But in order to allow extraction out of the complement clause at all, it is necessary to allow that to compose with a proper initial type of s. That is, we must countenance the inference:

\[
(\text{that}.x.y) \\
\hline
(\text{that}.x).y
\]

7. A partially associative version of the Lambek calculus.
Let Phon = \(\langle\text{Syl}^+, \{\}, (\cdot)\rangle\) be a structure on which two functions \(\{\cdot\}\) and \((\cdot)\) are defined. We use underlined letters \(a, b, c, \ldots\) as members of an infinite set of variables ranging over Syl+. We write \((a.b)\) and \((a.b)\) to indicate the operation of the functions \(\{\cdot\}\) and \((\cdot)\) on \(a\) and \(b\). The function \(\{\cdot\}\) is associative and thus satisfies:

\[
\{x.y\}.z = \{x.(y.z)\}
\]

The function \((\cdot)\) is not associative, but satisfies the two laws:

\[
(x.(y.z)) = ((x.y).z) \\
(x.(y.z)) = (((x.y).z)
\]

In addition, we impose the following requirement: the closure of the co-domain of \((\cdot)\) under the operation \(\{\cdot\}\) is contained in Syl+. This sub-structure of Syl+ is thus a semigroup.

Relative to a set P of primitive categories, the set of grammatical categories is defined inductively as the smallest set C such that
1. If \( x \) is in \( P \) and \( a \) is a variable ranging over \( \text{Syl}^+ \), then \(<x, a>\) is in \( C \).

2. If \(<x, a>\) and \(<y, b>\) are members of \( C \), then so are:

\[
\begin{align*}
&<x, a> \cdot <y, b>, (a, b) \\
&<x, a> \cdot <y, b>, (a, b) \\
&<x, (a, b) > / <y, b>, a \\
&<x, (a, b) > / <y, b>, a \\
&<y, b > \setminus <x, (b, a) >, a \\
&<y, b > \setminus <x, (b, a) >, a \\
\end{align*}
\]

Note that there are now six different categorial types, rather than the three forms of types in \( \text{AL} \): for each type in \( \text{AL} \), there are now two types, one containing the phonological form \((a, b)\) and one containing the phonological form \((a, b)\). The purpose of this contrast is to allow certain elements to introduce intrinsic bracketings. In the presentation of the axioms and inference rules below, we write \(<..., I_zI>\) for either \(<..., \{z\}>\) or \(<..., (z)>\). When we choose a given instantiation for one occurrence of square brackets in a rule, we must choose the same instantiation for all occurrence of square brackets within the rule.

We turn now to the formulation of axioms and rules of inference for this system, which we call \textit{partial AL}. We begin with a set of phonological axioms:

\[P_1. \quad (a.(b.c)) \iff_p ((a.b).c)\]

\[P_2. \quad (a.(b.c)) \iff_p ((a.b).c)\]

The first of these is simply the associativity axiom for \( (.) \). The second allows an element \( a \) which introduces non-associative bracketing to shrink the bracketing to an initial subsequence of an argument formed by the associative product.\(^6\)

The version of the Lambek system we propose has a single axiom and a number of inference rules. Upper-case letters are used as variables over the left projections of elements of \( C \):

\[A_1. \quad <x, [a]> \iff <x, [a]>\]

\[R_0. \quad [(x, y, z)] \iff [(x, y, z)] \]

\[R_1. \quad <<X.Y>, [x, y]> \iff <<Z, x, y>\]

\[R_1'. \quad <<Y.X>, [y, x]> \iff <<Z, y, x>\]

\[X \iff <<Z, [x, y]> / Y, x\]

\[X \iff <<Y \setminus Z, [y, x]>, x\]

76
The proof of rightward functional application then takes the following form:

\[
<Z, x> \rightarrow <Y, x> \rightarrow <Z, x> \quad (A1)
\]

This is valid whether we construe \([x, y]\) as \((x, y)\) or as \((x, y)\). From the conclusion of the above proof, by rule R1', we have:

\[
Y \rightarrow <Z, x> \rightarrow <Z, x> \quad (A1)
\]

This analogue of one of the two symmetrical forms of type-lifting is also valid regardless of how we construe \([x, y]\).

But composition and the other theorems that depend on it are not valid for all forms of types containing a product compatible with \([x, y]\). In particular, we have the following cases of rightward composition:

C1. \(<Z, z, y, v> / <Y, y, v>, z> \rightarrow <Z, z, y, v> / <Y, y, v>, z>

C2. \(<Z, z, y, v> / <Y, y, v>, z> \rightarrow <Z, z, y, v> / <Y, y, v>, z>

C1 covers two cases, depending on whether we interpret the expressions \([z, y, v]\) and \([z, y]\) as \((z, y, v)\) and \((z, y)\) or as \((z, y, v)\) and \((z, y)\). Both interpretations are valid (because of the role played in RO by the phonological axioms P1 and P2, respectively). But C2 is underivable in this system.

As a consequence, if we recharacterize the cases discussed earlier in the framework of partial AL, and if we assign the complementizer that to the category \(<c, (that, s)> / <s, g>, that>\), it will have exactly the properties necessary in this system to exhibit the behavior found in the paradigm with which we began. A rigorous demonstration of this fact exceeds the constraints on space imposed here, however, but will be treated in Oehrle (in preparation).

8. Concluding remarks.

As noted at the outset of our paper, there have been a number of attempts to characterize, in terms of "gaps" or "traces", the paradigm of cases with which we began. (We have made no attempt to catalogue them all.) What we have tried to do is show that there is an algebraic family of categorial systems that includes not only AL and nAL, but systems like partial AL, between them that provide a way of characterizing this paradigm with no reference to gaps or traces. The solution that we have proposed involves prohibiting certain classes
of expressions from engaging in the full range of bracketing relations allowed by the associativity axiom of AL. This same solution is equally applicable to reduced prepositions and the contracting forms of English aux-elements, although we cannot explore these consequences here. If the various contexts in which extraction is blocked can be shown to have common phonological properties, then we may interpret the operator ( . ) in Phon as correlating directly with these properties. In this case, our analysis provides a way of relating a class of extraction phenomena with other grammatical properties -- a step forward, we think, in comparison to theories that require ad hoc filters or constraints to characterize extractability. If these various contexts share no common phonological characterization, however, we may still construe the associative and non-associative modes of concatenation discussed here as abstract structures with properties that yield desirable consequences. The system partial AL allows the flexibility of AL to be combined judiciously with the rigidity of nAL in a way that provides an interesting alternative to accounts based on movement analyses, traces, and derivationally-based morphology of contracted forms.

Footnotes

1. For general background on categorial grammar, see Oehrle, Bach, and Wheeler (1988), Buszkowski, Marciszewski, and van Benthem (1988), and Moortgat (1989).

2. On the logical side, see van Benthem (1988) for an investigation of a permutation-closed variant of AL; on the linguistic side, note that the structural completeness of AL implies that in the system AL + Permutation which results from adding (7) to the axioms of AL, the GB rule "move alpha" is valid.

3. For alternative ways of treating discontinuous dependencies of this kind, see Moortgat (1989) and Pareschi (1988).

4. At least not so readily: such conjunctions are available on general grounds in the framework for conjunction of Oehrle (1987).

5. Our use of phonological variables has affinities with certain unification-based theories of grammatical composition, such as (Pollard & Sag, 1987) and Zeevat, Klein, and Calder (1987).

6. There is a failure of symmetry here: brackets do not care whether they are introduced by an element on the left of a string or by an element on the right, but axiom P2 is asymmetrical. In a more elaborate version, this symmetry is easily restored. But in the present context, since we will only be concerned with functors which introduce bracketing around an argument to the right (for example, proclitics), we ignore the required elaboration here.

References


van Benthem, J. 1988. The Lambek Calculus. Categorial Grammar and Natural


