PREDICTIVE NORMAL FORMS FOR FUNCTION COMPOSITION IN CATEGORIAL GRAMMARS

Robert E. Wall, University of Texas at Austin
and
Kent Wittenburg, MCC

Abstract: Extensions to Categorial Grammars proposed to account for nonconstitutent conjunction and long-distance dependencies introduce the problem of equivalent derivations, an issue we have characterized as spurious ambiguity from the parsing perspective. In Wittenburg (1987) a proposal was made for compiling Categorial Grammars into predictive forms in order to solve the spurious ambiguity problem. This paper investigates formal properties of grammars that use predictive versions of function composition. Among our results are (1) that grammars with predictive composition are in general equivalent to the originals if and only if a restriction on predictive rules is applied, (2) that modulo this restriction, the predictive grammars have indeed eliminated the problem of spurious ambiguity, and (3) that the issue of equivalence is decidable, i.e., for any particular grammar, whether one needs to apply the restriction or not to ensure equivalence is a decidable question.

1. **Introduction.** Steedman (1985, 1987), Dowty (1987), Moortgat (1988), Morrill (1988), and others have proposed that Categorial Grammar, a theory of syntax in which grammatical categories are viewed as functions, be generalized in order to analyze "noncanonical" syntactic constructions such as wh-extraction and nonconstituent conjunction. A consequence of these augmentations is an explosion of semantically equivalent derivations admitted by the grammar, a problem we have characterized as spurious ambiguity from the parsing perspective (Wittenburg 1986). In Wittenburg (1987), it was suggested that the offending rules of these grammars could take an alternate predictive form that would eliminate the problem of spurious ambiguity. This approach, consisting of compiling grammars into forms more suitable for parsing, is within the tradition of discovering normal forms for phrase structure grammars, and thus our title. Our approach stands in contrast to those which are attempting to address the spurious ambiguity problem in Categorial Grammars through the parsing algorithm itself rather than through the grammar (see Pareschi and Steedman 1987; Moortgat 1987, 1988; Hepple and Morrill 1989; Koenig 1989; Gardent and Bes 1989). Our approach is more in line with the tack that Bouma (1989) is taking, although his formulation of categorial systems differs radically from our own, more traditional set of assumptions.

In Wittenburg (1987) it was conjectured that predictive forms for Categorial Grammars were equivalent to the source forms and that they did indeed eliminate spurious ambiguity. Here we report on formal results that have ensued from these original conjectures. We have found that, on the whole, the conjectures proved valid although we have discovered that the relationship between predictive normal forms for these grammars and their source forms are more complicated than was implied by the earlier paper. As we will show, an additional condition is necessary to ensure equivalence of these grammars and eliminate spurious ambiguity from the picture.

2. **Source Grammar (G)** In this paper we focus on the role of basic function composition as a way of illustrating the effects of predictive normal forms. For these proofs then, we assume a form of Categorial Grammar that is considerably more restricted than those advocated by van Bentham (1986), Steedman (1987), Moortgat (1988), Morrill (1988), and others. As the work of these authors shows, the simple Categorial Grammars we assume here are not linguistically adequate. We do not consider the effects of type-raising nor of generalized conjunction here, nor do we address the issue of generalized composition. While we intend to address these points in future work, the simplifications we assume here allow us to uncover an intitial set of properties associated with the use of predictive combinators.

We assume for our source grammar G the following combinatory rules together with a lexically assigned system of categories of the usual recursive sort. That is, we assume a set of basic categories, say, {S, NP, N}. If X and Y are categories, so are X/Y and Y\X. Our notation follows Steedman (1987) and Dowty (1985) in that the domain type appears consistently to the right of a slash and a range type to the left. Left directionality is then indicated by a left-leaning slash, and right directionality by a right-leaning slash. Semantically, we assume that lexical categories introduce functional constants in lambda terms where the arity of the functions bears an obvious and direct relation to the syntactic type.[1] Here are example lexical entries.

kicks: S\NP/NP	John: S/(S\NP)	a: NP/N	platypus: N
λxλy ((**kicks** x) y)	λf(f **john**)	λx(**a** x)	**platypus**

We assume the following set of combinatory rules:

Forward function application (fa>)

X/Y Y -> X
f a f(a)

Backward function application (fa<)

Y X\Y -> X
a f f(a)

Forward function composition (fc>)

X/Y Y/Z -> X/Z
f g λx(f(g(x))) = Bfg

Backward function composition (fc<)

Y\Z X\Y -> X\Z
g f λx(f(g(x))) = Bfg

Given these semantics, G yields equivalence classes of derivations, where equivalence is defined modulo β-conversion of semantic terms.[2] The two sources of spurious ambiguity in G are summarized by the following equivalences generalized over directional variants of the rules:

[1] Although we use the term semantics here to describe the relevant issues of derivational ambiguity, it should be understood that we dealing with a syntactic domain. One might think of our semantics as defining the syntactic structures yielded by derivations using these grammars.

[2] This definition of equivalence does not take quantifier scope differences into account. It is more in harmony with the predictive normalization techniques to assume that scoping structure is not necessarily isomorphic to the derivation tree, a position also advocated by Steedman (1987) and Moortgat (1988).

(apply (compose X Y) Z) = (apply X (apply Y Z))

(compose X (compose Y Z)) = (compose (compose X Y) Z)

An example illustrating the first of these equivalences follows:[1]

```
        S                                     S
      f(g(a))                               f(g(a))
-------------------------fa>      -----------------------fa>
      S/NP                                  FVP
    λx(f(g(x)))                             g(a)
------------------fc>                  --------------fa>
S/FVP    FVP/NP      NP          S/FVP     FVP/NP    NP
 f          g         a            f         g         a
```

Assuming the terminal string "John kicks a platypus", complete derivations would yield the equivalent derivational terms ((**kicks** (**a platypus**))**John**).

The numbers of these equivalent derivations increase "almost exponentially" in string length, with the Catalan series (Wittenburg 1986).

3. **Predictive Normal Form (G')** A predictive normal form version of G replaces each composition rule with two predictive variants.[2]

Forward-predictive forward function composition (fpfc>)

```
X/(Y/Z)  Y/W -> X/(W/Z)
   f      g      λh(f(Bgh)) = λh(f(λx(g(h(x)))))
```

Backward-predictive forward function composition (bpfc>)

```
W/Z  X\(Y/Z) -> X\(Y/W)
 g      f        λh(f(Bhg))= λh(f(λx(h(g(x)))))
```

Backward-predictive backwards function composition (bpfc<)

```
Y\W   X\(Y\Z)  ->  X\(W\Z)
 g       f         λh(f(Bgh)) = λh(f(λx(g(h(x)))))
```

Forward-predictive backwards function composition (fpfc<)

```
X/(Y\Z)   W\Z  -> X/(Y\W)
   f       g       λh(f(Bhg))= λh(f(λx(h(g(x)))))
```

---

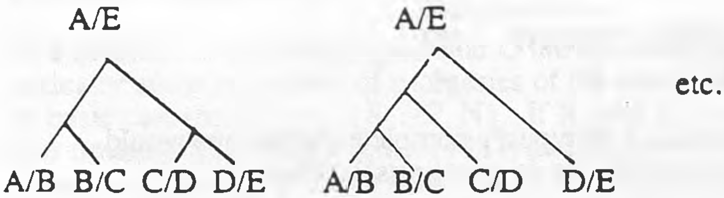
[1] FVP is used as a notational convenience for the category S\NP.

[2] These rules are derivable in the Lambek calculus (Lambek 1958).

We will now consider, first, the question of ambiguity in G'. Second, we will take up the question of whether G and G' are equivalent.

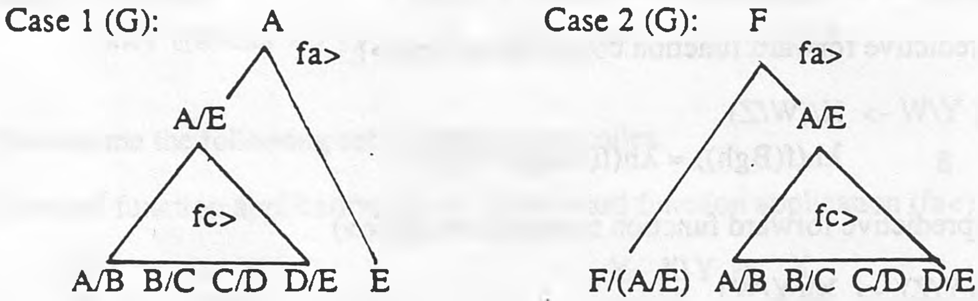
4. **Ambiguity in G'** Is there ambiguity in G'? We will consider first cases that are analogous to the derivations in G known to give rise to spurious ambiguity. Our proof is by induction on the height of a derivation tree.

In G, spurious ambiguity arises from the use of composition. Consider any maximal subtree of fc> in a derivation in G, i.e.,

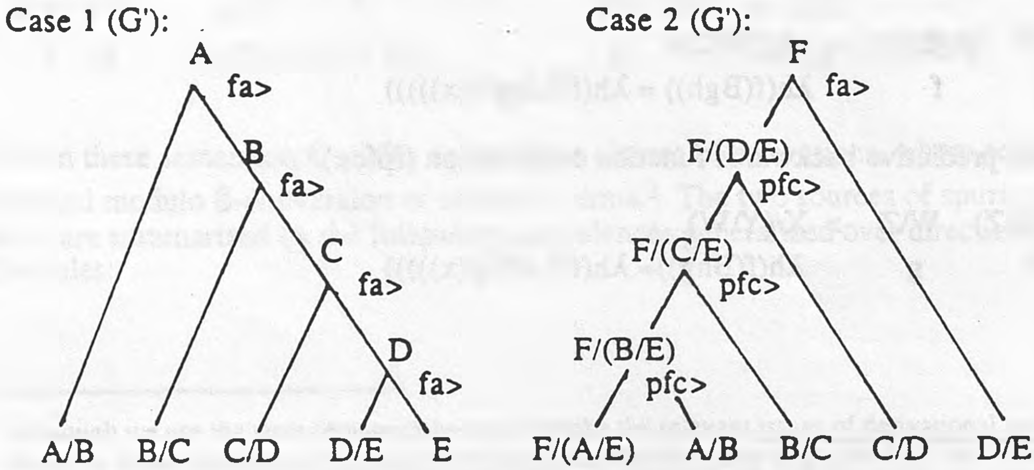


Since it is part of a derivation of S, it must feed into an instance of fa at the top (either as functor or as argument) -- if it fed into fc, this tree would not be a maximal fc tree.

So subderivations in G with fc> must be of one of the following forms:



In either case, there is one and only one derivation in G' for the same category sequence.



The cases of fc< are parallel. And since fc> and fc< cannot appear together in a maximal fc tree because of directionality clash, all cases are accounted for.

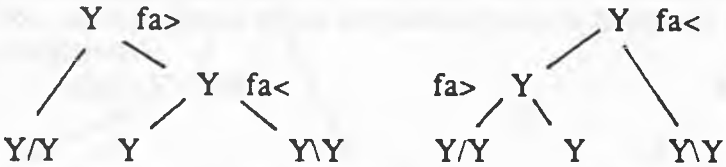
We have shown here that cases of spurious ambiguity in G do not give rise to analogous spurious ambiguity in G', but of course there may be new sources of ambiguity in G' that we have not yet considered.

Can there be any cases of derivational ambiguity in G'? That is, can there be derivation trees of the form

for (possibly complex) categories A, B, C, X, Y, Z, where mothers are derived from daughters using just the rules of fa and predictive function composition? An exhaustive list of all the combinatory possibilities reveals just two types:

Type I: X = Y/Y and Z = Y\Y

The central category Y can combine first by fa with Y/Y to its left or with Y\Y to its right, to yield Y in either case. This Y can then combine with the remaining category by fa to give Y again:

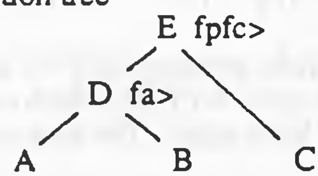


But this is a genuine ambiguity, not a spurious one, for the topmost Y can be assigned different semantic values by the two derivations. If [[Y/Y]] = f, [[Y]] = a, and [[Y\Y]] = g, the left derivation yields f(g(a)) and the right one g(f(a)).

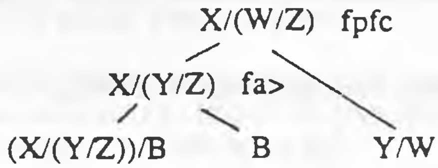
In the more general case, we might have m instances of Y/Y to the left of the Y and n instances of Y\Y to the right. In such a situation the number of syntactically and semantically distinct derivations would be the (m+n)th Catalan number. And since only fa> and fa< are used, the same ambiguity, if it is present, will be found in both G and in G'.

Type II: A predictive combination rule is involved in the derivation. We will illustrate with just one case; the others are similar, differing only the directions of the slashes and the order of constituents.

Consider the derivation tree

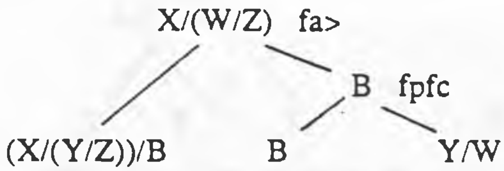


in which each mother node is derived from its daughters by the indicated rule. Since E is derived by fpfc>, D must be of the form X/(Y/Z) and C of the form Y/W; hence E is of the form X/(W/Z). Then because D is derived by fa>, it follows that A must be of the form (X/(Y/Z))/B. That is, the derivation tree is of the form

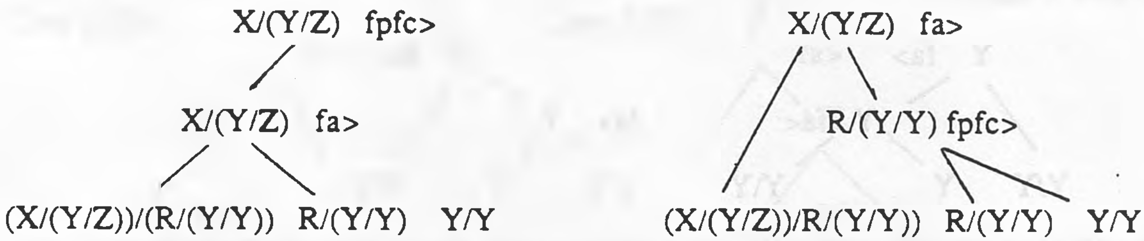


for (possibly complex) categories B, W, X, Y, Z.

Given the rules of fa and predictive composition, there is a distinct derivation tree yielding X/(W/Z) from the category sequence (X/(Y/Z))/B, B, Y/W; namely,



Now because (X/(Y/Z))/B becomes X/(W/Z) by fa>, it follows that X/(Y/Z) = X/(W/Z), and so Y = W. Further, B combines with Y/W (i.e., Y/Y) to give B again, so B is required to be of the form R/(Y/Y), for some R. (Note that R/(Y/Y) could also combine with Y/Y by fa>, but nothing prevents fpfc> from applying here as well.) In summary, G' allows the following sort of derivational ambiguity (and others symmetrical to it):



Is this a spurious or a genuine ambiguity? Letting the three leaf constituents have semantic values f, g, and h, respectively, we obtain λi[f(g)(Bhi)] for the root node of the left tree and f[λi[g(Bhi)]] for the root of the tree on the right. (Bhi denotes the composition of functions h and i.) These expressions are certainly non-equivalent for aribitrary functions f, g, h. [1] At any rate, we might ask if this sort of ambiguity can lead to an explosion of combinatorial possibilities like the one we were trying to rid ourselves of in the first place. The worst case would be when there is a sequence of n categories Y/Y extending rightward, thus:

(X/(Y/Z))/(R/(Y/Y))   R/(Y/Y)   Y/Y   Y/Y . . . Y/Y

Now R/(Y/Y) can combine with Y/Y's by fpfc, yielding R/(Y/Y) each time, then combine with the large category on the left by fa> to give X/(Y/Z), which can then combine with any remaining Y/Y's by fpfc> to give X/(Y/Z) back again. The lone instance of fa> can thus

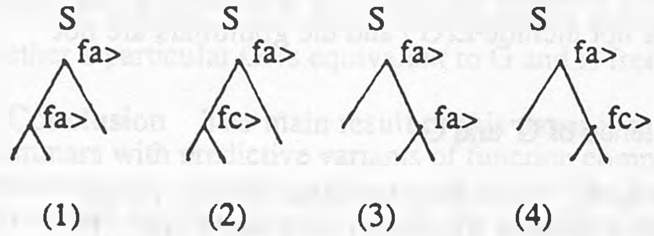
[1]Even so, it appears that if these functions are constrained by the form of the categories to which they are assigned (e.g., h must be a function from [[Y]]-type things to [[Y]]-type things, etc.), then the two expression may be equivalent and the ambiguity is a "spurious" one in the language of G'. At any rate, this point is moot given succeeding comments that these derivations need to be ruled out for G' to be equivalent to G.

occur at any point in the derivation, and if there are n Y/Y's, there will be n+1 distinct derivation trees. Thus, the number of derivations grows only linearly with the number of occurrences of Y/Y, not with a Catalan growth rate.

5. Equivalence of G and G' In considering equivalence of these grammars, we first take up the question of whether L(G) is a subset of L(G') followed by the question of whether L(G') is a subset of L(G).

5.1. <u>Predictive composition includes composition</u> Proof sketch: We show by induction on the depth of derivation trees that any derivation in G has a derivation in G'.

Any derivation of category S in G must end in fa> (or fa<). Consider the extension by depth one of a derivation tree headed by fa>. We consider 4 (not always mutually exclusive) cases. (Others include the symmetrical < variants and those that are excluded by directionality clashes).



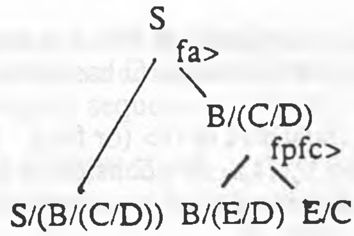
Cases (1) and (3) are common to G and G'. Consider case (2). From the definitions of fa> and fc>, the categories of the derivation must be as shown on the left, where Y and Z are any categories.

In G' there is a corresponding derivation from the same sequence of categories, as shown on the right. There is also this derivation in G, but G', lacking fc>, has only this one for this category sequence.

Consider case (4).

G' lacks fc>, but fpfc> allows (just) one derivation for this category sequence. The other cases symmetrical to these follow similarly.

5.2. Does L(G) subsume L(G')? Consider the following derivation in G':



There is no corresponding derivation in G. (Neither fa> nor fc> is applicable to the given categories.) Thus, in general, L(G) does not include L(G') and the grammars are not equivalent.

What can be done about the non-equivalence of G' and G?

1. **Restrict rule application in G'**: One may stipulate that the result category of a predictive rule cannot serve as argument in any other rule. (In function application X/Y Y => Z we take Y to be the argument category. In predictive rule X/(Y/Z) Y/W => X/(W/Z) we take the Y/W to be the argument. For backwards rules, the argument category is the leftmost term.) In the derivation just above, the predictive rule fpfc> "feeds" fa> as argument. If derivations in G' are restricted in this way, L(G') is provably included in L(G), and the grammars are weakly equivalent.[1]

Moreover, the same restriction banishes all cases of Type II ambiguity noted in Sec. 4 above. Observe that Type II ambiguity depends on predictive rules in G' being able to "feed" the arguments of further instances of predictive rules. Thus, G' becomes free of any spurious ambiguity.

This approach might be thought to be reminiscent of Pareschi and Steedman (1987), where spurious ambiguity is addressed through procedural means in parsing. Yet our approach here actually need not constrain the parsing algorithm at all. A node formed by a predictive rule can be flagged, say, by a feature, while those formed by fa would not be. All combinatory rules could then have a feature on their "argument" categories that would block when encountering this flag. This rather minimal amount of additional bookkeeping could easily be accommodated in the parsing strategy of one's choice: top-down, bottom-up, left-right, breadth-first, or whatever. Thus, what at first might appear to be a constraint on parsing would be more accurately described as a modification to the grammar.

2. **Grin and bear it**: Recasting the grammar in "predictive normal form" eliminates all cases of spurious ambiguity occasioned by sequences of function composition, a problem which is known to crop up very frequently in actual

---

[1] For lack of space, we do not include the full proof here. It is parallel to the proof in Sec. 5.1 showing the inclusion of L(G) in L(G'). Any derivation in this newly restricted G' is provably replacable by a derivation in G.

applications and to cause serious delays in parsing times. On the other hand, because of the complexity and the rather specific forms of the categories which give rise to the spurious ambiguities and the "spurious derivations" in the G' examples above, it seems reasonable to suppose that such cases are unlikely to be encountered very often in ordinary applications. In any event, as we noted above, the number of Type II ambiguous derivations in G' grows only linearly and not in Catalan fashion with increasing string length and would not be expected to lead to intolerable parsing times. The slight profligacy of G' over G might, therefore, present no serious practical problem.

For those still inclined to worry, we offer the following reassuring fact: a predictive normal form grammar can misbehave only if categories of sufficient "complexity" can be derived from the given set of categories in the lexicon, e.g., a category of the form S/(X/(W/Z)) in the case of non-equivalence above and of the form (X/(Y/Z))/(R/(Y/Y)) in the instances of Type II ambiguity. But given such a grammar and the lexical categories it is a decidable question whether any categories of the undesired complexity can arise during a derivation.[1] (We wish to thank Jim Barnett for suggestions on how to prove this.) Thus one can tell whether a particular G' is equivalent to G and is free from spurious ambiguity.[2]

6. Conclusion The main result of this paper is that we have shown that Categorial Grammars with predictive variants of function composition rules can satisfy the requirements for normalization, namely, that the "compiled" grammars preserve equivalence and that they do so with the benefit of eliminating the parsing problem occasioned by spurious ambiguity. We have also enumerated decidability proofs of interest. Our next task is to explore the predictive normal form strategy with more expressive, and more nearly adequate, Categorial systems such as those that incorporate some form of generalized composition and conjunction, type-raising, etc. What we expect to find is that if predictive normalization techniques are applicable at all, the predictive grammars will have a relationship to their source forms that parallels the one we have uncovered here. In other words, we expect the restriction on the use of predictive rules is in general necessary for preserving equivalence when using predictive combinators.

7. Acknowledgements This work has been carried out under the auspices of the Interface Languages project in the Human Interface Laboratory of the Advanced Computer Technology Program at MCC.

8. References

Bouma, G. (1989) Efficient Processing of Flexible Categorial Grammar. In Proceedings of the Fourth Conference of the European Chapter of the Association for Computational Linguistics, , 10-12 April 1989, pp. 19-26

Dowty, D. (1987) Type Raising, Functional Composition, and Non-Constituent Conjunction. In Oehrle, R., E. Bach, and D. Wheeler (eds.), Categorial Grammars and Natural Language Structures. Dordrecht: Reidel.

---

[1] The proof of these decidability results is contained in a longer version of this paper (MCC technical report ACT-HI-274-89) available from MCC, Human Interface Lab, 3500 W. Balcones Research Center Drive, Austin, TX 78759.

[2] N.B. Type-raising does increase complexity of categories in a different way, and thus these observations do not extend to categorial grammars with such rules (e.g., Moortgat 1987).

Gardent, C., and G. Bes. 1989. Efficient Parsing for French. In Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics, 26-29 June 1989, Vancouver, pp. 280-287.

Hepple, M., and G. Morrill. (1989) Parsing and Derivational Equivalence. In Proceedings of the Fourth Conference of the European Chapter of the Association for Computational Linguistics, 10-12 April 1989, Manchester, England, pp. 9-18.

Koenig, E. 1989. Parsing as Natural Deduction. In Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics, 26-29 June 1989, University of British Columbia, Vancouver, pp. 272-286.

Lambek, J. (1958). The Mathematics of Sentence Structure. American Mathematical Monthly 65:154-170.

Moortgat, M. (1987) Lambek Categorial Grammar and the Autonomy Thesis. Paper presented at the ZWO Symposium 'Morphology and Modularity', Utrecht, 16-18 June 1986. [Available as INL Working Paper 87-03, Instituut voor Nederlandse Lexicologie, Leiden, Netherlands.]

Moortgat, M. (1988) Categorial Investigations: Logical and Linguistic Aspects of the Lambek Calculus. Foris.

Morrill, G. (1988) Extraction and Coordination in Phrase Structure Grammar and Categorial Grammar. Ph.D. dissertation, Centre for Cognitive Science, University of Edinburgh.

Pareschi, R., and M. Steedman (1987) A Lazy Way to Chart-Parse with Categorial Grammars. Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics, 6-9 July 1987, Stanford, pp. 81-88.

Steedman, M. (1985) Dependency and Coordination in the Grammar of Dutch and English. Language 61:523-568.

Steedman, M. (1987) Combinatory Grammars and Parasitic Gaps. Natural Language and Linguistic Theory 5:403-440.

Van Bentham, J. (1986) Essays in Logical Semantics. Reidel.

Wittenburg, K. (1986) Natural Language Parsing with Combinatory Categorial Grammars in a Graph-Unification-Based Formalism. Ph.D. dissertation, University of Texas at Austin.

Wittenburg, K. (1987) Predictive Combinators: A Method for Efficient Parsing of Combinatory Categorial Grammars. Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics, 6-9 July 1987, Stanford, pp. 73-80.