Categorial Grammar and Lexical-Functional Grammar*

Reinhard Muskens
Department of Linguistics
Tilburg University
The Netherlands

Proceedings of the LFG01 Conference
University of Hong Kong, Hong Kong
Miriam Butt and Tracy Holloway King (Editors)
2001
CSLI Publications
http://csli-publications.stanford.edu/

*I wish to thank the participants of LFG2001, especially Mary Dalrymple and Ron Kaplan, for valuable feedback. All comments are very welcome.

Abstract

This paper introduces λ-grammar, a form of categorial grammar that has much in common with LFG. Like other forms of categorial grammar, λ-grammars are multi-dimensional and their components are combined in a strictly parallel fashion. Grammatical representations are combined with the help of *linear combinators*, closed pure λ-terms in which each abstractor binds exactly one variable. Mathematically this is equivalent to employing linear logic, in use in LFG for semantic composition, but the method seems more practicable.

While λ-grammars could be used to formalize many approaches to grammatical theory, they are certainly natural as a basis for the formalization of LFG. This leads to a theory I would like to call λ-LFG. In this paper it will be shown how the standard components of LFG can be set up in the framework. We will have descriptions of c-structure, descriptions of f-structure, and semantics. The difference between defining and constraining information will be explained in terms of entailment, and requirements on long-distance paths in f-structure will be explained in terms of entailment in the presence of a simple set of axioms.

1 Introduction

In this paper I want to discuss a version of Lexical-Functional Grammar (LFG) that is also a version of Categorial Grammar (CG). A convergence between the two frameworks has set in at least since Zeevat, Klein, and Calder (1986) proposed a 'sign-based' categorial grammar and Oehrle (1988) defined 'multi-dimensional' categorial grammars. The sign-based approach, now adopted by many categorial grammarians (e.g. Moortgat 1991; Morrill 1994; Moortgat 1997),[1] allows various grammatical representations to be combined in tandem, a move clearly reminiscent of LFG's multi-component architecture.

A second important step leading to further convergence of the two frameworks was Dalrymple et al.'s (1993) proposal to use Linear Logic[2] as a 'glue' between f-structure and semantics. The $\{\multimap, \otimes\}$ fragment of Intuitionistic Linear Logic that is used in this 'glue' approach is in fact identical with the undirected Lambek Calculus.[3] This still leaves a gap between the two frameworks, as most versions of

[1]Signs in Categorial Grammar are sequences of representations. For example, Zeevat, Klein, and Calder (1986) work with signs of the form ⟨phonology, syntactic category, semantics, order⟩. Each slot in the sequence may be associated with its own form of reasoning. This should be distinguished from the concept of a sign in Head-Driven Phrase Structure Grammar, where multiple levels of the grammar are represented in one attribute-value matrix.

[2]The original paper on Linear Logic is (Girard 1987). An attractive textbook (Troelstra 1992).

[3]For the original version of the Lambek Calculus, see Lambek 1958; for the undirected version

Categorial Grammar are directional and distinguish between categories A/B (seeking a B to the right) and $B\backslash A$ (seeking a B to the left). Linear Logic and the undirected Lambek Calculus collapse this to a single $B \multimap A$, a formula that consumes a B and then produces an A, irrespective of order. The motivation for using a directional system comes from a wish to treat word order directly on the level of the calculus. Grammatical formalisms that treat word order in a different way, as LFG does, can make do with nondirectionality.

In fact the treatment of word order can be factored out of the general calculus in CG as well. Consider the signs in (1), which are in the style of (Moortgat 1991; Morrill 1994) and are triples consisting of a string, a semantic term, and a directional category. Since (1a) has a type which seeks an *np* to the left, it can be combined with (1b). The combination consists of string concatenation in the first dimension (forming *likes John*), application in the second dimension (forming $\lambda x \lambda y.[like(y,x)](j)$, which reduces to $\lambda y.like(y,j)$), and a form of Modus Ponens in the third dimension. The result is as in (1c).

(1) a. likes : $\lambda x \lambda y.like(y,x) : (np\backslash s)/np$

b. John : $j : np$

c. likes John : $\lambda y.like(y,j) : np\backslash s$

The directionality of the types here codes the way in which strings can be concatenated, but, as was observed in (Oehrle 1994; Oehrle 1995), this information can be shifted to the syntactic dimension if we are willing to let syntactic representations be lambda terms, in analogy with semantic representations. (1a) then becomes (2a), whose first element is a lambda term over strings and whose last element is an undirected type. Combining (2a) with (1b), now using application in the first as well as in the second dimension, leads to (2b).

(2) a. $\lambda \mathrm{x} \lambda \mathrm{y}$. y likes x : $\lambda x \lambda y.like(y,x) : np \multimap (np \multimap s)$

b. $\lambda \mathrm{y}$. y likes John : $\lambda y.like(y,j) : np \multimap s$

There is much to be said for such a shift of word order information from the level of types to the syntactic dimension.[4] A priori it seems that once we have a syn-

Benthem 1986; Benthem 1988; Benthem 1991; for a survey of Categorial Grammar in the Lambek tradition Moortgat 1997.

[4]Interestingly, Curry (1961) already argues for this approach. Curry considers *functors*, which are expressions containing subscripted blanks, such as '$—_1$ is between $—_2$ and $—_3$' or '$—_1$ were eaten by the children'. Functors can apply to arguments and arguments are to be substituted for blanks in the order of the subscripts. Essentially then, although Curry does not explicitly mention this, functors are lambda terms over syntactic objects. For example, the first of the functors just mentioned can also be written '$\lambda x \lambda y \lambda z.x$ is between y and z'.

tactic dimension, word order should be handled there. But there are also empirical consequences, one positive, the other less so. Let us look at a consequence that is less than positive. One clear attraction of categorial grammars is the way in which (non-constituent) coordination is treated (see e.g. Steedman 1985; Dowty 1988; Moortgat 1988). The sign in (3a) can be formed from (1a) and (1b) with the help of hypothetical reasoning and can subsequently be coordinated with the similar sign in (3b). The result is as in (3c), which may be used to obtain (3d). Coordination is concatenation in the first dimension (with the coordinating word in between the coordinated elements) and either union or intersection in the semantic dimension, depending on whether the coordination is a disjunction or a conjunction. The types of the coordinated signs must be equal and the result will have the same type again.

(3) a. John likes : $\lambda x.like(j,x) : s/np$

b. Mary hates : $\lambda x.hate(m,x) : s/np$

c. John likes but Mary hates : $\lambda x.like(j,x) \wedge hate(m,x) : s/np$

d. John likes but Mary hates bananas : $like(j,b) \wedge hate(m,b) : s$

Treatments of coordination along these lines are not without their problems and usually result in overgeneration (see (Milward 1994; Moortgat 1997) for good discussions), but nevertheless seem to fare well compared to many other approaches. The analysis depends on a reduction of the syntactic dimension to *strings* and collapses when richer structures such as trees are taken to be basic. A move to representations such as the ones in (2) does not seem to be compatible with this analysis of coordination either.

On the other hand such representations immediately solve one of the problems of the standard Lambek Calculus. Since *John likes* can be analysed as an s/np, it is easy to obtain parses for sentences such as (4a) in Lambek's system. The trick is to categorize *who* as an item that searches an s/np to its right. *who* must also be categorized as searching for an $np\backslash s$ in view of (4b). But what to do if the gap is medial, as in (4c)?

(4) a. a woman who John likes

b. a woman who likes John

c. a woman who John likes enormously

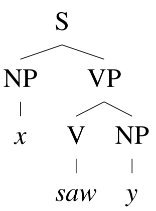
In order to deal with this problem clever extensions of the Lambek Calculus have been proposed (e.g. Morrill 1994). But these extensions are also considerable complications of the original idea. For the kind of representations in (2), on the other hand, a problem with medial gaps does not even arise: The relevant representations can be taken to be λx.John likes x, λx.x likes John, and λx.John likes x enormously, all of type $np \multimap s$. The relative pronoun can subcategorize for this type.[5]

(5) a. Everyone loves someone

b. Bill thinks someone kissed Carol

Similarly, it is well known that the Lambek Calculus can deal with scope ambiguities as long as the scope taking elements are in a peripheral position: (5a) will get its desired two readings. But as soon as a scope taking element is non-peripheral, difficulties may arise (Hendriks 1993; Morrill 1994; Moortgat 1997; Dalrymple et al. 1999). For example, (5b) will not obtain a reading in which *someone* takes scope over *thinks*. Again, operators (such as Moortgat's ↑) have been introduced that cleverly circumvent the problem (see Dalrymple et al. 1999 however) but also complicate the categorial machinery. Moving to an undirected calculus will make the problem disappear, as the periphery does not play any special role in such a system. I take this to be strong evidence that word order information should not be treated on the level of the type system but should be dealt with in a syntactic dimension. If it is represented in the type system this essentially syntactic information causes trouble in the semantics.

If Oehrle's move from simple strings to lambda-ed strings and from a directed to an undirected calculus is accepted then a third step narrowing the gap between CG and LFG is made. The same calculus now plays a central role in both grammatical frameworks. An obvious next step, made in (Oehrle 1999), is to bring the various components of the categorial grammar into line with the components that are recognized in LFG. Oehrle considers sequences such as the one in (6), consisting of a lambda-ed c-structure, a lambda-ed f-structure, a semantic term, and a type. Such signs are combined with the help of the undirected Lambek Calculus.

(6) $\lambda yx.$ 

$$: \lambda yx. \begin{bmatrix} \text{PRED} & \text{`SEE'} \\ \text{TENSE} & \text{PAST} \\ \text{SUBJ} & x \\ \text{OBJ} & y \end{bmatrix} \; : \; \lambda yx.see(x,y) \; : \; np \multimap (np \multimap s)$$

[5]The sign for the pronoun *who* could be taken to be λXλy.y who X(ε) : $\lambda X \lambda Y \lambda z.X(z) \wedge Y(z)$: $(np \multimap s) \multimap (n \multimap n)$, where ε is the empty string.

syntax:	$k, f : \nu$	$t, F : \nu t$	$T, \mathcal{F} : (\nu t)(\nu t)$	
semantics:	$x, y : e$	$i, j : s$	$p : st$	$P : e(st)$

Table 1: Typographical conventions for variables used in this paper. *Var* : *Type* means that *Var* (with or without subscripts or superscripts) always has type *Type*.

Oehrle's work will be my point of departure and should receive much praise, but I have a point of criticism as well. The terms in the first two dimensions of (6) are *structures*. More precisely, they are functions that take certain arguments and then give structures as a result. While I think it possible to flesh this out formally, I also think it is not very promising as a formalization of LFG. One of the important insights of (Kaplan and Bresnan 1982) was a separation, at least in the f-dimension, of *descriptions* and the structures satisfying them. This distinction, which is also at the heart of model theory, is lost if signs get a form as in (6).[6] As a consequence, it will not be possible to model *disjunctive* or *negative* constraints. It will also not be possible to formalize the difference between *defining* and *constraining* information, or to explain path constraints on f-structure arising out of long-distance dependencies. Such mechanisms are part and parcel of Lexical-Functional Grammar and in this paper I will show that they can easily be accounted for within a combined CG-LFG approach if a shift from structures to descriptions is made.

A second departure from Oehrle's proposal that I want to suggest involves a simplification of the technical machinery which now becomes possible. Thus far we have considered signs that needed to be combined with the help of some form of Lambek Calculus, but it is well-known (see Benthem 1986, for example) that proofs in the implicational fragment of the undirected Lambek Calculus (= Linear Logic) are isomorphic with *linear combinators*, closed pure λ-terms in which each abstractor binds exactly one free variable. This will make it possible to do away with proofs altogether and to just consider certain linear combinations of lexical elements.

2 λ-Grammars

To make concrete what I have in mind, I will define a small toy grammar in this section. In the next section some elaborations and revisions will be discussed. Consider the three signs in (7).

(7) a. john : $\lambda f.arc(f, cat, N) \wedge arc(f, num, sg) \wedge arc(f, pers, 3) : john$

[6] A footnote in (Oehrle 1999) credits a referee with suggesting a move from structures to descriptions in the f-dimension.

b. $\mathsf{mary} : \lambda f.arc(f,cat,N) \wedge arc(f,num,sg) \wedge arc(f,pers,3) : mary$

c. $\lambda t_1 \lambda t_2.[t_2\ [\mathsf{loves}\ t_1]] :$
$\lambda F_1 \lambda F_2 \lambda f \exists f_1 f_2 [F_1(f_1) \wedge F_2(f_2) \wedge arc(f,cat,V) \wedge arc(f,tense,pres) \wedge$
$arc(f_1,cat,N) \wedge arc(f,obj,f_1) \wedge arc(f_2,cat,N) \wedge arc(f_2,num,sg) \wedge$
$arc(f_2,pers,3) \wedge arc(f,subj,f_2)] :$
$\lambda x \lambda y \lambda i.love(y,x,i)$

These signs each consist of a c-structure component, an f-structure component, and a semantic component. Expressions in sans serif in the c-structure terms are of type νt, and denote sets of nodes.[7] For example, john can be thought of as the set of nodes that are labeled 'John', whereas an expression such as $[\mathsf{loves}\ \mathsf{mary}]$ can be thought of as the set of nodes k directly dominating a node k_1 labeled 'loves' and a node k_2 labeled 'Mary', with k_1 preceding k_2. More information about c-structure components will follow in the next section.

The f-components of our signs consist of λ-terms over the first order feature language of (Johnson 1991) and the semantics in the third component is in accordance with a streamlined form of Montague's (1973) theory. Constants *john* and *mary* are of type e and *love* is of type $e(e(st))$. Constants *cat, num, pers*, etc. are of a type a (attributes), while N, sg, $3, \ldots$ are of type ν (nodes).[8] More typing information is given in Table 1. For the moment, we consider a grammar with three dimensions, but in general the number of dimensions of a grammar is arbitrary (though fixed). The terms that we are interested in are all closed and we require that lexical elements have closed terms in each dimension.

Signs can be combined by means of *pointwise application*. In general, if $M = \langle M_1, \ldots, M_n \rangle$ and $N = \langle N_1, \ldots, N_n \rangle$ are sequences of λ-terms such that $M_i(N_i)$ is well-typed for each i, the pointwise application of M to N is just

$$\langle M_1(N_1), \ldots, M_n(N_n) \rangle \ .$$

Generalizing the notation for application, we denote this as $M(N)$. It is easily seen that the result of pointwise application of (7c) to (7a) equals (8a) modulo standard equivalences and that (8a)((7b)) reduces to (8b).

(8) a. $\lambda t_2.[t_2\ [\mathsf{loves}\ \mathsf{john}]] :$
$\lambda F_2 \lambda f \exists f_1 f_2 [F_2(f_2) \wedge arc(f,cat,V) \wedge arc(f,tense,pres) \wedge arc(f_1,cat,N)$
$\wedge\, arc(f,obj,f_1) \wedge arc(f_2,cat,N) \wedge arc(f_2,num,sg) \wedge arc(f_2,pers,3) \wedge$
$arc(f,subj,f_2)] :$
$\lambda y \lambda i.love(y,john,i)$

[7] We drop the $\multimap$ in types, in conformity with the usage in semantics.

[8] For simplicity, we make no type distinction between tree nodes and feature nodes in this paper, but the conceptual distinction is important of course.

b. [mary [loves john]] :
$\lambda f \exists f_1 f_2 [arc(f,cat,V) \wedge arc(f,tense,pres) \wedge arc(f_1,cat,N) \wedge arc(f,obj,f_1) \wedge arc(f_2,cat,N) \wedge arc(f_2,num,sg) \wedge arc(f_2,pers,3) \wedge arc(f,subj,f_2)]$:
$\lambda i.love(mary,john,i)$

The three descriptions in sentential signs such as (8b) each denote a set in every possible model of the language; the first two sets of nodes (type νt), the third a set of possible worlds (a proposition, type *st*). The idea is that if the second set is non-empty in some model of the axioms in Johnson 1991 (see below), then any node satisfying the first description should be connected to the truth conditions expressed in the third element. The requirement that the second component should be satisfiable provides for a subcategorization mechanism. E.g., combining (8a) with a plural subject would have led to an f-description that can only denote the empty set.

In (9) and (10) some more lexical signs are given with two results of their possible combinations in (11).

(9) a. man : $\lambda f.arc(f,cat,N) \wedge arc(f,num,sg) \wedge arc(f,pers,3)$: *man*

b. woman : $\lambda f.arc(f,cat,N) \wedge arc(f,num,sg) \wedge arc(f,pers,3)$: *woman*

(10) a. $\lambda t \lambda T.T[$a $t]$:
$\lambda F \lambda \mathcal{F}.\mathcal{F}(\lambda f.F(f) \wedge arc(f,cat,N) \wedge arc(f,num,sg) \wedge arc(f,pers,3))$:
$\lambda P' P \lambda i \exists x[P'(x)(i) \wedge P(x)(i)]$

b. $\lambda t \lambda T.T[$every $t]$:
$\lambda F \lambda \mathcal{F}.\mathcal{F}(\lambda f.F(f) \wedge arc(f,cat,N) \wedge arc(f,num,sg) \wedge arc(f,pers,3))$:
$\lambda P' P \lambda i \forall x[P'(x)(i) \rightarrow P(x)(i)]$

(11) a. $\lambda T.T[$every man$]$:
$\lambda \mathcal{F}.\mathcal{F}(\lambda f.arc(f,cat,N) \wedge arc(f,num,sg) \wedge arc(f,pers,3))$:
$\lambda P \lambda i \forall x[man(x,i) \rightarrow P(x)(i)]$

b. $\lambda T.T[$a woman$]$:
$\lambda \mathcal{F}.\mathcal{F}(\lambda f.arc(f,cat,N) \wedge arc(f,num,sg) \wedge arc(f,pers,3))$:
$\lambda P \lambda i \exists x[woman(x,i) \wedge P(x)(i)]$

The terms that our signs consist of are typed, but it is expedient to type the signs themselves as well. Types for signs will be called *abstract types*. Abstract types in

abstract type	syntactic dimensions	semantic dimension
S	νt	st
NP	νt	e
N	νt	$e(st)$

Table 2: Concretizations of abstract types used in this paper.

this paper are built up from ground types S, NP and N with the help of implication, and thus have forms such as NP S, N((NP S)S), etc. A restriction on signs is that a sign of abstract type A should have a term of type A^i in its i-th dimension. The values of the function $.^i$ for ground types can be chosen on a per grammar basis and in this paper are as in Table 2. For complex types, the rule is that $(AB)^i = A^iB^i$. This means, for example, that $\text{NP(NP S)}^1 = \text{NP(NP S)}^2 = (\nu t)((\nu t)\nu t)$ and that $\text{NP(NP S)}^3 = e(e(st))$. As a consequence, (7c) should be of type NP(NP S). Similarly, (7a) and (7b) can be taken to be of type NP, (8a) and (8b) are of types NP S and S respectively, etc. In general, if M has abstract type AB and N abstract type A, then the pointwise application $M(N)$ is defined and has type B.

Abstraction can also be lifted to the level of signs. Supposing that the variables in our logic have some fixed ordering and that the number of dimensions of the grammar under consideration is n, we define the k-th n-dimensional variable ξ of abstract type A as the sequence of variables $\langle \xi_1, \ldots, \xi_n \rangle$, where each ξ_i is the k-th variable of type A^i. The *pointwise abstraction* $\lambda\xi M$ is then defined as $\langle \lambda\xi_1 M_1, \ldots, \lambda\xi_n M_n \rangle$. A definition of *pointwise substitution* is left to the reader.

With the definitions of pointwise application, pointwise abstraction, and n-dimensional variable in place, we can consider complex terms built up with these constructions. (12a), for example, is the pointwise application of (11b) to the pointwise composition of (11a) and (7c). Here ζ is of type NP. (12a) can be expanded to (12b), where each dimension of a lexical sign is denoted with the help of an appropriate subscript (e.g. $(11b)_1$ is $\lambda T.T[\textsf{a woman}]$). The terms here can be reduced and the result is as in (12c), a sign coupling the c-description in its first dimension to one of its possible readings. The other reading is obtained from (12d), which reduces to (12e).

(12) a. $(11b)(\lambda\zeta.(11a)((7c)(\zeta)))$

b. $(11b)_1(\lambda\zeta_1.(11a)_1((7c)_1(\zeta_1)))$:
$(11b)_2(\lambda\zeta_2.(11a)_2((7c)_2(\zeta_2)))$:
$(11b)_3(\lambda\zeta_3.(11a)_3((7c)_3(\zeta_3)))$

c. [[every man] [loves [a woman]]] :
$\lambda f \exists f_1 f_2[arc(f, cat, V) \wedge arc(f, tense, pres) \wedge arc(f_1, cat, N) \wedge$

$arc(f, obj, f_1) \wedge arc(f_2, cat, N) \wedge arc(f_2, num, sg) \wedge arc(f_2, pers, 3) \wedge$
$arc(f, subj, f_2)]$:
$\lambda i \exists y[woman(y,i) \wedge \forall x[man(x,i) \rightarrow love(x,y,i)]]$

d. $(11a)(\lambda\zeta_2.(11b)(\lambda\zeta_1.(7c)(\zeta_1)(\zeta_2)))$

e. [[every man] [loves [a woman]]] :
$\lambda f \exists f_1 f_2[arc(f, cat, V) \wedge arc(f, tense, pres) \wedge arc(f_1, cat, N) \wedge$
$arc(f, obj, f_1) \wedge arc(f_2, cat, N) \wedge arc(f_2, num, sg) \wedge arc(f_2, pers, 3) \wedge$
$arc(f, subj, f_2)]$:
$\lambda i \forall x[man(x,i) \rightarrow \exists y[woman(y,i) \wedge love(x,y,i)]]$

Let us call terms such as (12a) and (12d), which are built up from lexical signs with the help of n-dimensional variables, pointwise application and abstraction, *n-terms*. It is worth to note that n-terms are subject to the laws of α, β, and η-conversion, i.e. reasoning with them is as usual. But clearly, not every n-term makes for an acceptable coupling between syntax and semantics. We restrict ourselves to *linear combinations* of lexical elements. These are n-terms that are closed and conform to the condition that every abstractor $\lambda\zeta$, with ζ an n-dimensional variable, binds exactly one free ζ. n-terms conforming to this condition are called *generated signs*.[9] Conditions such as the requirement that the third component of a generated sign must be satisfiable are *admissibility* conditions and a generated sign obeying them is called *admissable*.

Multidimensional grammars that are set up in the way sketched here, with λ-terms in each dimension of the grammar and linear combination as a generative device, will be called *λ-grammars*. If a λ-grammar is meant to be an alternative formalization of LFG insights, as the grammars in this paper will be, it is called a λ-LFG. For more information on λ-grammars, see (Muskens 2001a).

Since any n-term M obeys the usual laws of λ-conversion, it can be written in the form $C(L_1)\cdots(L_m)$, where $L_1,\ldots,L_m$ are lexical signs and C is an n-term that does not contain any lexical material. If M is closed, C is a multi-dimensional (and typed) variant of a *combinator* in the sense of (Curry and Feys 1958). In case M is a generated sign, C will correspond to a *linear* (or **BCI**) combinator. For example, (12a) can be rewritten as (13), with $\lambda Q_1 \lambda R \lambda Q_2.Q_1(\lambda\zeta.Q_2(R(\zeta)))$ playing the role of the linear combinator combining (11b), (7c), and (11a).

(13) $\lambda Q_1 \lambda R \lambda Q_2.Q_1(\lambda\zeta.Q_2(R(\zeta)))((11b))((7c))((11a))$

From the fact that linear combinators play an important underlying role we see that λ-grammars have obvious affinities not only with LFG and Lambek Categorial Grammar, but also with Combinatory Categorial Grammar (see e.g. Steedman

[9]Note that any linear combination of generated signs is itself a generated sign.

1996; Steedman 2000). But λ-grammars should be distinguished from standard categorial grammars in that they are non-directional and do not use derivations.

3 λ-LFG

The purpose of this section is twofold. First we need to fill in some details that were left open in the definition of our toy grammar. We will take a closer look at the c-description and f-description components. For the logic of the semantic component the reader is referred to the first chapters of (Muskens 1995). When the necessary details have been filled in, we will show how the descriptions approach that is taken in this paper allows for the incorporation of some further ideas that are central to LFG.

3.1 A Closer Look at C-descriptions and F-descriptions

3.1.1 C-descriptions

Terms such as [loves [a woman]] in fact can be taken to be abbreviations of tree descriptions. We flesh this out by providing the ν domain with binary relations $\lhd^+$ (proper dominance), $\lhd$ (immediate dominance), and $\prec$ (precedence) and by imposing the necessary structure by means of axioms (see also Cornell 1994; Backofen et al. 1995; Muskens 2001b). Here we just adopt the requirements in (14). (14a) states that the relations $\lhd^+$ and $\prec$ are strict partial orders, while (14b) and (14c) impose *Inheritance*, (14d) requires *Rootedness* (r is a constant of type ν here), and (14e) defines $\lhd$ as an immediate dominance relation in terms of $\lhd^+$.[10] The last axiom excludes the possibility that leaf nodes have more than one label.

(14) a. $\lhd^+$ and $\prec$ are irreflexive and transitive

b. $\forall k_1 k_2 k_3\,[[k_1 \lhd^+ k_2 \wedge k_1 \prec k_3] \rightarrow k_2 \prec k_3]$

c. $\forall k_1 k_2 k_3\,[[k_1 \lhd^+ k_2 \wedge k_3 \prec k_1] \rightarrow k_3 \prec k_2]$

d. $\forall k\,[r \lhd^+ k \vee r = k]$

e. $\forall k_1 k_2\,[k_1 \lhd k_2 \leftrightarrow \forall k_3\,[k_1 \lhd^+ k_3 \lhd^+ k_2 \rightarrow [k_3 = k_1 \vee k_3 = k_2]]]$

f. $\forall k \neg[\delta_1(k) \wedge \delta_2(k)]$, if δ_1 and δ_2 are distinct lexical labels

[10] These requirements in themselves do not suffice to axiomatize the notion of linguistic tree. For instance, the usual requirement of *Exhaustivity* ($\forall k_1 k_2\,[k_1 \prec k_2 \vee k_2 \prec k_1 \vee k_1 \lhd^+ k_2 \vee k_2 \lhd^+ k_1 \vee k_1 = k_2]$) is not met. But the axioms are sufficient in the sense that the terms we generate will have a minimal model which is a tree.

Using the relations $\lhd$ and $\prec$, we define $[A_1 \cdots A_m]$ to be an abbreviation of (15).[11]

(15) $\lambda k \exists k_1 \ldots k_m [A_1(k_1) \wedge \ldots \wedge A_m(k_m) \wedge k \lhd k_1 \wedge \ldots \wedge k \lhd k_m \wedge k_1 \prec \cdots \prec k_m]$

In other words, $[A_1 \cdots A_m]$ will denote the set of nodes with daughters $A_1, \ldots, A_m$ (in that order), as expected. E.g., [loves [a woman]] now is short for (16).

(16) $\lambda k \exists k_1 k_2 [\mathsf{loves}(k_1) \wedge k \lhd k_1 \wedge k \lhd k_2 \wedge k_1 \prec k_2 \wedge \exists k_3 k_4 [\mathsf{a}(k_3) \wedge \mathsf{woman}(k_4) \wedge k_2 \lhd k_3 \wedge k_2 \lhd k_4 \wedge k_3 \prec k_4]]$

From (16) we see that a minimality requirement is needed: As things stand the statement $[A_1 \cdots A_m](k)$ may hold in some model in which k has daughters other than $A_1, \ldots, A_m$.[12] We exclude this possibility by our interpretation of generated signs. First, define the relation $\sqsubset$ between models for the tree language by letting $M' \sqsubset M$ if and only if (a) the domain of M' is a proper subset of the domain of M, (b) the root of M' is the root of M, and (c) elements of M' are in the proper dominance relation in M' if and only if they are in the proper dominance relation in M. (There is no similar constraint on precedence.) Now let $\mathcal{S}$ be a generated admissable sign with c-description component C and semantic component S. We say that a tree model M *expresses* S if M is a model of the axioms in (14) that satisfies $C(r)$ and no $M' \sqsubset M$ is both a model of the axioms in (14) and satisfies $C(r)$. The minimality condition with respect to $\sqsubset$ serves to rule out structures M that have more nodes than were intended.

Thus far, we have not put any category information into our c-descriptions. This could be done very easily by letting category labels such as AP, VP, ... be terms of type ν and by stipulating that $[_L A_1 \cdots A_m]$, where L is a category label, is short for $\lambda k(arc(k, cat, L) \wedge [A_1 \cdots A_m](k))$. However, we shall prefer treating major category information on a par with feature information in the rest of this paper.

Writing $[A_1 \cdots A_m]$ for (15) will be handy in circumstances where we want the grammar to prescribe the way in which the daughters of a given mother are ordered. But this is not always what we want. In many languages the order of daughters is not rigidly constrained. The non-configurational Warlpiri language (Simpson 1991), for example, has essentially free word order in nominal and verbal finite clauses, the only restriction being that 'auxiliaries' always take second position.[13]

[11] The use of square brackets in $[A_1 \cdots A_m]$ is special and should be distinguished from its normal use in terms.

[12] One way to rule out such undesired structures would be to strengthen the definition of $[A_1 \cdots A_m]$ with an extra condition of the form $\forall k'[k \lhd k' \to [k' = k_1 \vee \ldots \vee k' = k_m]]$. However, such a strengthening would preclude the possibility of attaching extra daughters to a given node while constructing a description.

[13] AUX elements with a polysyllabic base can take first or second position (Simpson 1991). I will ignore this possibility for the sake of exposition.

In (17) it is illustrated how word order in the simple sentence *Ngarrka-ngku ka wawirri panti-rni* can vary, as long as the present tense 'auxiliary' is in second position. (The example is from (Hale 1983). We limit ourselves to three of the six possible permutations.)

(17) a. Ngarrka-ngku ka wawirri panti-rni
man-ERG PRES kangaroo(ABS) spear-NPST
The man is spearing the kangaroo

b. Wawirri ka ngarrka-ngku panti-rni

c. Panti-rni ka wawirri ngarrka-ngku

d. *Panti-rni wawirri ka ngarrka-ngku

This behaviour can be modeled by weakening the description in (15) and requiring pairwise disequalities instead of a series of precedences. Let us write $[A_1 \cdots A_m]^u$ for the following property:

(18) $\lambda k \exists k_1 \ldots k_m [A_1(k_1) \wedge \ldots \wedge A_m(k_m) \wedge k \triangleleft k_1 \wedge \ldots \wedge k \triangleleft k_m \wedge \bigwedge_{i \neq j} k_i \neq k_j]$

$[A_1 \cdots A_m]^u(k)$ will hold if k is the mother of daughters $A_1, \ldots, A_m$; but this time no ordering is prescribed.

We need a way to model the fact that AUX elements always take second position. Let us write A^{snd} for:

(19) $\lambda k [A(k) \wedge \exists k_1 k_2 [k_1 \triangleleft k \wedge k_1 \triangleleft k_2 \wedge k_2 \prec k \wedge \forall k_3 [[k_1 \triangleleft k_3 \wedge k_3 \prec k] \rightarrow k_2 = k_3]]]$

The term in (20) will now describe all grammatical variations of sentence (17) in the sense that any of its minimal models is an acceptable tree for one of these variations while each acceptable tree is a model.

(20) $[\textsf{Panti-rni wawirri ka}^{\text{snd}}\ \textsf{ngarrka-ngku}]^u$

3.1.2 F-descriptions

The following three axioms are a direct adaptation from (Johnson 1991). The first puts a functionality requirement on the transition relation. The second embodies the constraint that atomic features have no further attributes (C_{val} stands for the set of constants denoting atomic features, such as *sg, past, ...*). And the third axiom schema gives constant-constant clashes by requiring that *past* $\neq$ *sg*, *past* $\neq$ *pres*, $V \neq N$, etc.

(21) a. $\forall a \forall f_1 f_2 f_3 [[arc(f_1, a, f_2) \wedge arc(f_1, a, f_3)] \rightarrow f_2 = f_3]$

b. $\forall a \forall f \neg arc(c, a, f)$, where $c \in C_{val}$

c. $c \neq c'$, for all syntactically distinct pairs $c, c' \in C_{val}$

For more information about these axioms the reader is referred to (Johnson 1991).

3.2 Incorporating More Ideas from LFG

3.2.1 Checking as Entailment

Thus far, information in the f-components of our signs has been of the *defining* kind, but LFG has always distinguished between defining and *constraining* equations (see Kaplan and Bresnan 1982). The first give positive information, the second an obligation to check whether certain information is present. For example, the defining equation $(\uparrow \text{INF}) = +$ sets the INF value of $\uparrow$ to $+$, whereas the constraining $(\uparrow \text{INF}) =_c +$ checks whether the INF value of $\uparrow$ is $+$ after all defining equations have been processed.

An obvious way to model this logically is to say that constraining information must be *entailed* by defining information. In (22) two of the signs in (7) are repeated, but this time there are *two* f-components. The first consists of defining information, the second of defining plus constraining material.

(22) a. john :
$\lambda f.arc(f, cat, N) \wedge arc(f, num, sg) \wedge arc(f, pers, 3)$:
$\lambda f.arc(f, cat, N) \wedge arc(f, num, sg) \wedge arc(f, pers, 3)$:
john

b. $\lambda t_1 \lambda t_2.[t_2\ [\text{loves}\ t_1]]$:
$\lambda F_1 \lambda F_2 \lambda f \exists f_1 f_2 [F_1(f_1) \wedge F_2(f_2) \wedge arc(f, cat, V) \wedge arc(f, tense, pres) \wedge arc(f, obj, f_1) \wedge arc(f, subj, f_2)]$:
$\lambda F_1 \lambda F_2 \lambda f \exists f_1 f_2 [F_1(f_1) \wedge F_2(f_2) \wedge arc(f, cat, V) \wedge arc(f, tense, pres) \wedge arc(f_1, cat, N) \wedge arc(f, obj, f_1) \wedge arc(f_2, cat, N) \wedge arc(f_2, num, sg) \wedge arc(f_2, pers, 3) \wedge arc(f, subj, f_2)]$:
$\lambda x \lambda y \lambda i.love(y, x, i)$

Note that the third element of (22b) contains more material than the second. The extra statements are the ones that need to be checked. In order for this to work we need to revise the definition of *admissability* slightly. Suppose that some set of axioms such as the ones in (21) is given. A generated sign of type S is now defined to be admissable if (a) its first f-component has a non-empty denotation in some model of the axioms, and (b) the denotation of its first f-component is a subset of the denotation of its second f-component in all models of the axioms. The second condition is in fact an entailment requirement.[14] The reader will have no difficulty in seeing that agreement is now enforced. In 'John loves Mary' the required entailment holds, but only because the f-descriptions for 'John' and 'Mary' provide the material requested by 'loves'.

There is a lot of duplication in the four-dimensional signs in (22) and in practice it seems possible to use the notation in (23) where just one term is used, with subscripts c on some subformulas. This is just short for a four-dimensional sign. The second f-description is the one that is given (without the subscripts), but the first is obtained by deleting all subscripted material. The checking process now boils down to the requirement that subscripted material must become redundant under the usual rules of logic.

(23) a. john : $\lambda f.arc(f,cat,N) \wedge arc(f,num,sg) \wedge arc(f,pers,3)$: $john$

b. $\lambda t_1 \lambda t_2.[t_2$ [loves t_1]] :
$\lambda F_1 \lambda F_2 \lambda f \exists f_1 f_2 [F_1(f_1) \wedge F_2(f_2) \wedge arc(f,cat,V) \wedge arc(f,tense,pres) \wedge$
$arc(f_1,cat,N)_c \wedge arc(f,obj,f_1) \wedge arc(f_2,cat,N)_c \wedge arc(f_2,num,sg)_c \wedge$
$arc(f_2,pers,3)_c \wedge arc(f,subj,f_2)]$:
$\lambda x \lambda y \lambda i.love(y,x,i)$

The formalization of constraining information given here is straightforward and in the end we employ a notation that is very close to the standard one. But note that explaining the distinction between defining and constraining material as that between what is given and what must be derived essentially requires working on a level of descriptions. A structural approach, as the one in (Oehrle 1999), cannot explain the difference in this way.

3.2.2 The Long Distance: Checking F-paths

The technique developed in the previous section can also be used to characterize non-local dependencies by checking whether certain types of path exist in the functional domain. Standard LFG uses the device of *functional uncertainty* (Kaplan and Zaenen 1989) here, but it seems that the present approach can handle path checking without additional machinery. Consider Kaplan and Zaenen's rule for English topicalization in (24).

[14] It may be worthy of notice that while the treatment of agreement in (7) rested crucially on the axioms in (21), the new treatment in (22) does not in fact make use of these axioms. The situation is discussed and explained in (Johnson 1999). We like to remain agnostic about the question whether in the end axioms such as those in (21) will be needed.

(24) $S' \quad \rightarrow \quad \underset{\substack{(\uparrow\text{TOPIC}) = \downarrow \\ (\uparrow\text{TOPIC}) = (\uparrow\{\text{COMP, XCOMP}\}^*(\text{GF} - \text{COMP}))}}{\text{XP or } S'} \quad S$

The second annotation on this rule states that there must be a path from the f-structure connected to the mother S′ to that connected to the topicalized element and that the path should, top-down, start with zero or more COMP or XCOMP transitions, and then bottom out with one transition labeled by a grammatical function (SUBJ, OBJ, OBJ2,...) other than COMP. It is not stated how many open or closed complements are present, whence the uncertainty. The series of complements is called the *body* of the path, the final grammatical function its *bottom*.

Suppose we wanted to prove in our system that a certain path was a topicalisation path (for English). Then the following axioms would be needed.

(25) a. $\forall f f' f''[[body^{\mathrm{T}}(f, f'') \wedge bottom^{\mathrm{T}}(f'', f')] \rightarrow path^{\mathrm{T}}(f, f'')]$

b. $\forall f\, body^{\mathrm{T}}(f, f)$

c. $\forall f f' f''[body^{\mathrm{T}}(f, f') \wedge arc(f', comp, f'')] \rightarrow body^{\mathrm{T}}(f, f'')]$
$\forall f f' f''[body^{\mathrm{T}}(f, f') \wedge arc(f', xcomp, f'')] \rightarrow body^{\mathrm{T}}(f, f'')]$

d. $\forall f f'[arc(f, subj, f') \rightarrow bottom^{\mathrm{T}}(f, f')]$
$\forall f f'[arc(f, obj, f') \rightarrow bottom^{\mathrm{T}}(f, f')]$, etc., for all grammatical functions save *comp*.

The first two of these Horn statements are not particular to English and say that a topicalisation path must consist of a body and a bottom and that a body may be empty. The statements in (25c,d), on the other hand, give the particular form that bodies and bottoms in English topicalization paths must have.

In order to be able to give a treatment of topicalization using these axioms, we must at least have one complement taking verb at our disposal. In (26) a sign for *thinks* is given, very much along lines that are now familiar.[15] In combination with similar signs (26) can provide the arbitrarily long distances that some syntactic constructions can bridge.

(26) $\lambda t_1 \lambda t_2.[t_2\ [\textsf{thinks}\ t_1]]$:
$\lambda F_1 \lambda F_2 \lambda f \exists f_1 f_2 [F_1(f_1) \wedge F_2(f_2) \wedge arc(f, cat, V) \wedge arc(f, tense, pres) \wedge$
$arc(f_1, cat, V)_c \wedge arc(f, comp, f_1) \wedge arc(f_2, cat, N)_c \wedge arc(f_2, num, sg)_c \wedge$
$arc(f_2, pers, 3)_c \wedge arc(f, subj, f_2)]$:
$\lambda p \lambda y \lambda i \forall j[B(y, j, i) \rightarrow p(j)]$

[15]The term in the semantic component of *thinks* gives a version of Hintikka's theory of belief as truth in all doxastic alternatives. Read $B(y, j, i)$ as 'world j is a doxastic alternative for y in world i'.

The sign in (27) gives a treatment of noun phrase topicalization. It has a form that requires it to be combined with (a) a sentence lacking a noun phrase which may be a generalized quantifier (type ((NP S)S)S), and (b) the lacking noun phrase (type (NP S)S). We use Z for variables of the first type, Q for those of the second. When applied to signs of such types (27) returns a sign of type S, so that its overall type is (((NP S)S)S)(((NP S)S)S).[16] The term in the first dimension of (27) preposes the syntactic material of its quantifier argument to the result of providing its first argument with a trace e. The term in the second dimension provides the Z argument with a new quantifier that essentially consists of the old one plus (a) the information that the f-structure connected with the NP is a topic of the f-structure of the result, and (b) the requirement that a topicalisation path must run from the latter to the former. The term in the semantic dimension merely copies the semantics of the Z argument.[17]

(27) $\lambda Z_1 \lambda Q_1.[Q_1(\lambda t.t)\ \ Z_1(\lambda T.T(\mathsf{e}))]$:
$\lambda Z_2 \lambda Q_2 \lambda f.$
$Z_2(\lambda \mathcal{F}.\mathcal{F}(\lambda f'.Q_2(\lambda F.F)(f') \wedge arc(f, topic, f') \wedge path^{\mathrm{T}}(f, f')_c))(f)$:
$\lambda Z_3.Z_3$

As an example of how this works, consider the generated sign in (28a), which can be shown to reduce to the sign in (28b). Many requirements that have already been checked are omitted in (28b), but the crucial requirement $path^{\mathrm{T}}(f, f_1)_c$ is displayed. However, this statement can easily be derived using $arc(f, comp, f_3)$, $arc(f_3, obj, f_1)$, and the axioms in (25). It is clear that these path axioms will always allow the requirement to be satisfied as long as the defining information provides a path of the right kind.

(28) a. $(27)\Big(\lambda Q.(26)\Big(Q(\lambda\zeta.(23b)(\zeta)((7b)))\Big)\Big((7a)\Big)\Big)\Big((11b)\Big)$

b. [[a woman][john[thinks[mary[loves e]]]]] :
$\lambda f \exists f_1 f_2 f_3 f_4 [arc(f_1, cat, N) \wedge arc(f_1, num, sg) \wedge arc(f_1, pers, 3) \wedge$
$arc(f, topic, f_1) \wedge path^{\mathrm{T}}(f, f_1)_c \wedge arc(f_2, cat, N) \wedge arc(f_2, num, sg) \wedge$
$arc(f_2, pers, 3) \wedge arc(f_3, cat, V) \wedge arc(f_3, tense, pres) \wedge arc(f_3, obj, f_1) \wedge$
$arc(f_3, subj, f_3) \wedge arc(f_4, cat, N) \wedge arc(f_4, num, sg) \wedge arc(f_4, pers, 3) \wedge$

[16] It seems we need to go this high in order to preserve scope possibilities. A simplified version of the sign makes do with type (NP S)(NP S):
$\lambda T \lambda t.[t\ T(\mathsf{e})] : \lambda \mathcal{F} \lambda F \lambda f.\mathcal{F}(\lambda f'.F(f') \wedge arc(f, topic, f') \wedge path^{\mathrm{T}}(f, f')_c)(f) : \lambda P.P$
This is simpler, but requires the NP to be quantified-in at the point of topicalization or higher.

[17] There are of course semantic and pragmatic consequences of topicalisation, but we ignore them here.

$$arc(f, cat, V) \wedge arc(f, tense, pres) \wedge arc(f, comp, f_3) \wedge arc(f, subj, f_4)] :$$
$$\lambda i \forall j[B(john, j, i) \rightarrow \exists x[woman(x, j) \wedge loves(mary, x, j)]]$$

We conclude that no extra mechanism is needed for path constraints. Path requirements are just constraining information in our approach and constraining information is modeled using entailment. But while the treatment of path requirements reduces to the treatment of constraints in general, there is still a computational difference between them and the simple agreement requirements that were met before. While the latter can be shown to be satisfied on the basis of simple properties of conjunctions, the former need reasoning in a Horn theory.

4 Conclusion

We have defined λ-grammars, a form of categorial grammar based on multidimensional signs that can be combined using linear combinators. We also hope to have shown that this form of grammar squares well with the set-up of LFG and many important ideas underlying that grammatical theory. Components of signs in λ-grammars are combined in a strictly parallel way. The linear combinators combining them provide a form of resource-sensitivity that is also present in the mathematically related linear logic. This takes care of the *Coherence* and *Completeness* requirements in LFG (approaches to LFG based on linear logic also take care of these requirements automatically). Since the various components of our grammar are terms, and since terms *describe* structure instead of providing it, we immediately reap some of the benefits of the descriptions approach to grammar. There is no difficulty with expressing negative or disjunctive information and it was shown that the distinction between defining and constraining information, important in LFG, can naturally be modeled as an entailment requirement. Constraints on long-distance paths can be modeled in essentially the same way, using a simple set of axioms characterizing acceptable paths.

The strictly parallel character has advantages, but it also has drawbacks. An advantage which the approach shares with other formalizations of LFG is that semantics need *not* be compositional with respect to surface structure. As I hope to show in a longer version of this paper, examples refuting surface compositionality, such as discontinuous adjective-noun combinations in Warlpiri and other non-configurational languages, pose no problem. A disadvantage of the strict parallelism in this paper is that the lack of communication between various components is total. Researchers in LFG have found it fruitful to mix statements about precedence and grammatical function (Bresnan 1995), but this is impossible in the present set-up. One way to allow a form of communication would be to let the various components share certain constants, but discussion about how this can be done in a sufficiently restricted way should also await the longer version of this paper.

References

Backofen, R., J. Rogers, and K. Vijay-Shankar (1995). A First-Order Axiomatization of the Theory of Finite Trees. *Journal of Logic, Language and Information 4*, 5–39.

Benthem, J. v. (1986). *Essays in Logical Semantics*. Dordrecht: Reidel.

Benthem, J. v. (1988). The Semantics of Variety in Categorial Grammar. In W. Buszkowski, W. Marciszewski, and J. v. Benthem (Eds.), *Categorial Grammar*, pp. 37–55. Amsterdam: John Benjamins.

Benthem, J. v. (1991). *Language in Action*. Amsterdam: North-Holland.

Bresnan, J. (1995). Linear Order, Syntactic Rank, and Empty Categories: On Weak Crossover. In M. Dalrymple, R. Kaplan, J. Maxwell III, and A. Zaenen (Eds.), *Formal Issues in Lexical-Functional Grammar*, Chapter 8, pp. 241–274. Stanford: Stanford University.

Cornell, T. (1994). On Determining the Consistency of Partial Descriptions of Trees. In *Proceedings of ACL-94*.

Curry, H. (1961). Some Logical Aspects of Grammatical Structure. In *Structure of Language and its Mathematical Aspects: Proceedings of the Twelfth Symposium in Applied Mathematics*, pp. 56–68. AMS.

Curry, H. and R. Feys (1958). *Combinatory Logic*, Volume I. Amsterdam: North-Holland.

Dalrymple, M., J. Lamping, F. Pereira, and V. Saraswat (1999). Quantification, Anaphora, and Intensionality. In M. Dalrymple (Ed.), *Semantics and Syntax in Lexical Functional Grammar*, Chapter 2, pp. 39–89. Cambridge, MA: MIT Press.

Dalrymple, M., J. Lamping, and V. Saraswat (1993). LFG Semantics via Constraints. In *Proceedings of the Sixth Meeting of the European ACL*. European Chapter of the Association for Computational Linguistics.

Dowty, D. (1988). Type-raising, Functional Composition, and Non-Constituent Conjunction. In R. Oehrle, E. Bach, and D. Wheeler (Eds.), *Categorial Grammars and Natural Language Structures*, pp. 153–197. Dordrecht: Reidel.

Girard, J.-Y. (1987). Linear Logic. *Theoretical Computer Science 50*, 1–102.

Hale, K. (1983). Warlpiri and the Grammar of Non-configurational Languages. *Natural Language and Linguistic Theory 1*(1), 5–47.

Hendriks, H. (1993). *Studied Flexibility: Categories and Types in Syntax and Semantics*. Ph. D. thesis, University of Amsterdam.

Johnson, M. (1991). Logic and Feature Structures. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, Sydney, Australia.

Johnson, M. (1999). Type-driven Semantic Interpretation and Feature Dependencies in R-LFG. In M. Dalrymple (Ed.), *Semantics and Syntax in Lexical Functional Grammar*, Chapter 10, pp. 359–388. Cambridge, MA: MIT Press.

Kaplan, R. and J. Bresnan (1982). Lexical-Functional Grammar: a Formal System for Grammatical Representation. In J. Bresnan (Ed.), *The Mental Representation of Grammatical Relations*, pp. 173–281. Cambridge, MA: The MIT Press.

Kaplan, R. and A. Zaenen (1989). Long-distance Dependencies, Constituent Structure, and Functional Uncertainty. In M. Baltin and A. Kroch (Eds.), *Alternative Conceptions of Phrase Structure*, pp. 17–42. Chicago: Chicago University Press.

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

Milward, D. (1994). Non-Constituent Coordination: Theory and Practice. In *Proceedings of the 15th International Conference on Computational Linguistics, Coling 94*, Kyoto, Japan, pp. 935–941.

Montague, R. (1973). The Proper Treatment of Quantification in Ordinary English. In *Formal Philosophy*, pp. 247–270. New Haven: Yale University Press.

Moortgat, M. (1988). *Categorial Investigations: Logical and Linguistic Aspects of the Lambek Calculus*. Ph. D. thesis, University of Amsterdam.

Moortgat, M. (1991). Labelled Deductive Systems for Categorial Theorem Proving. In P. Dekker and M. Stokhof (Eds.), *Proceedings of the Eighth Amsterdam Colloquium*, Amsterdam, pp. 403–423.

Moortgat, M. (1997). Categorial Type Logics. In J. v. Benthem and A. t. Meulen (Eds.), *Handbook of Logic and Language*, pp. 93–177. Elsevier.

Morrill, G. (1994). *Type Logical Grammar: Categorial Logic of Signs*. Dordrecht: Kluwer.

Muskens, R. (1995). *Meaning and Partiality*. Stanford: CSLI.

Muskens, R. (2001a). Language, Lambda's, and Logic. Submitted for publication.

Muskens, R. (2001b). Talking about Trees and Truth-conditions. *Journal of Logic, Language and Information 10*(4), 417–455.

Oehrle, R. (1988). Multi-Dimensional Compositional Functions as a Basis for Grammatical Analysis. In R. Oehrle, E. Bach, and D. Wheeler (Eds.), *Categorial Grammars and Natural Language Structures*, pp. 349–389. Dordrecht: Reidel.

Oehrle, R. (1994). Term-Labeled Categorial Type Systems. *Linguistics and Philosophy 17*, 633–678.

Oehrle, R. (1995). Some 3-Dimensional Systems of Labelled Deduction. *Bulletin of the IGPL 3*, 429–448.

Oehrle, R. (1999). LFG as Labeled Deduction. In M. Dalrymple (Ed.), *Semantics and Syntax in Lexical Functional Grammar*, Chapter 9, pp. 319–357. Cambridge, MA: MIT Press.

Simpson, J. (1991). *Warlpiri Morpho-Syntax: A Lexicalist Approach*, Volume 23 of *Studies in Natural Language and Linguistic Theory*. Dordrecht: Kluwer.

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

Steedman, M. (1996). *Surface Structure and Interpretation*. MIT Press.

Steedman, M. (2000). *The Syntactic Process*. MIT Press.

Troelstra, A. (1992). *Lectures on Linear Logic*. Stanford: CSLI.

Zeevat, H., E. Klein, and J. Calder (1986). Unification Categorial Grammar. Manuscript.